Oracle® HTTP Server

Administering a Standalone Deployment Based on Apache 2.0

10*g* Release 2 (10.1.2)

B14009-02

July 2005

ORACLE®

Oracle HTTP Server Administering a Standalone Deployment Based on Apache 2.0, 10*g* Release 2 (10.1.2)

B14009-02

Primary Author: Harry Schaefer

Contributing Author: Julia Pond, Sanket Atal, Warren Briese, Olivier Caudron, Kevin Clark, Priscila Darakjian, Sander Goudswaard, Helen Grembowicz, Mathew Joy, Pushkar Kapasi, Keith Kelleman, Eric Kienle, John Lang, Bruce Lowenthal, Li Ma, Chuck Murray, Mark Nelson, Carol Orange, Bert Rich, Jon Richards, Shankar Raman, Baogang Song, Kevin Wang, Karen Wilson

Contributor:

Contents

7 Understanding Modules

E Third Party Licenses

Glossary

Index

Preface

This guide describes how to administer Oracle HTTP Server.

Audience

Oracle Application Server Administering a Standalone Deployment Based on Apache 2.0 is intended for application server administrators, security managers, and managers of databases used by application servers.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, see the Oracle Application Server Documentation Library.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Overview

This chapter describes the Oracle HTTP Server, highlighting the differences between the Oracle distribution and the open source Apache product on which it is based. It also explains how to start, stop, and restart the server.

Topics discussed are:

Oracle HTTP Server Features

Oracle HTTP Server is the Web server component of Oracle Application Server. Based on the **Apache** infrastructure, Oracle HTTP Server allows developers to program their site in a variety of languages and technologies - Perl (through `mod_perl` and CGI), C (through CGI, and FastCGI), C++ (through FastCGI), PHP, and Oracle's PL/SQL. It can also be a proxy server, both forward and reverse. In addition, the features of single sign-on, clustered deployment, and high availability, enhances the operations of Oracle HTTP Server.

Based on Apache - HTTP v1.1 Support

Oracle HTTP Server code is based on Apache 2.0 Web Server (`http://www.apache.org`). With such a proven code base, Oracle HTTP Server provides Oracle Application Server customers with the stability, flexibility, and scalability required of a Web server.

Security - Encryption with SSL

Secure Sockets Layer is required to run any Web site securely. Oracle HTTP Server supports SSL encryption based on industry standard, patented, algorithms. The SSL works seamlessly with both Internet Explorer and Netscape browsers. In addition, the infrastructure has been upgraded to share the same wallet information as the database users. Features include:

- **SSL HW Acceleration Support**: SSL encryption is slower when done in software. Dedicated hardware support for this purpose is now supported, specifically with nCipher.
- **Variable Security per Directory**: This feature allows individual directories to be protected by different strength encryption.

- **Oracle HTTP Server to OC4J SSL Support**: Oracle HTTP Server and OC4J can communicate using AJP protocol over SSL. Previously, Oracle HTTP Server and OC4J used the AJP 1.3 protocol unencrypted, without support for authentication. Now, Oracle HTTP Server has been modified to extend support to the AJP 1.3 protocol over SSL providing both encryption and authentication.

  See Also:

  - *Oracle Application Server Security Guide*
  - Chapter 8, "Managing Security"
  - Chapter 9, "Enabling SSL for Oracle HTTP Server"

Security - Single Sign On

Oracle HTTP Server supports the standard basic authentication features of Web servers. The source for the username and password used here is a flat file (with encrypted passwords). In addition, a module, `mod_osso`, is included to support single sign on across sites and across applications. This provides for a much better end user experience (they have to login only once), and a much easier development cycle (most of the security is declarative).

See Also:

- *Oracle Application Server Single Sign-On Administrator's Guide*
- "mod_osso" on page 7-18

Virtual Hosts

The virtual host facility allows an HTTP server to service multiple domain names over one IP address. Thus, virtual hosts `www.north.com` might have the same IP address as `www.south.com`. Oracle HTTP Server provides a "container" environment for a virtual host, thus providing a virtual host with its own set of security and other configuration directives, in addition to locations from which the files are served. This allows an ISP to save on hardware and administrative costs by enabling hundreds to thousands of sites to be served from a single runtime instance of Oracle HTTP Server. Only one virtual host on a single IP address can accommodate SSL. Oracle HTTP Server can support multiple IP addresses and each one of them can have one, but only one virtual host.

Proxy Server and URL Rewriting

Any Web site that is "alive" changes often. Along with that, the directory structure and the URLs change. Oracle HTTP Server makes it easy to accommodate these changes by including an engine that support URL rewriting so that the end users do not have to change their bookmarks. It also supports reverse proxy capabilities, thus making it easier to make content served by different servers to appear from one single server.

Server Side Include

Server Side Includes provide an easy way of adding some dynamic, or uniform static content, across all the site's pages. It is typically used for header/footer information. Oracle HTTP Server supports special directives to enable these only for certain types of files, or for given virtual hosts.

Perl

Perl is a scripting language often used to provide dynamic content. Perl can either be called as a CGI program or directly through `mod_perl`. Oracle Application Server uses Perl version 5.8.3.

> **See Also:** "mod_perl" on page 7-18

PHP

PHP is an open source, widely-used, general-purpose, client-side scripting language, that is embedded in standard HTML. It is used to generate dynamic HTML pages.

> **See Also:** "mod_php" on page 7-20

C/C++ (CGI and FastCGI)

CGI programs have been commonly used to program Web applications. Oracle HTTP Server improves on them by providing a mechanism to keep them alive beyond the request lifecycle, thus improving the performance tremendously.

Dynamic Monitoring Service

Dynamic Monitoring Services (DMS) metrics give runtime performance statistics for both Oracle HTTP Server and OC4J processes. As applications run, DMS collects detailed performance statistics. This data enables you to monitor the duration of important request processing phases and status information. With this information, you can locate performance bottlenecks and tune the application server to maximize throughput and minimize response time.

> **See Also:** *Oracle Application Server Performance Guide*

Oracle Process Manager and Notification Server

Oracle Application Server provides a high availability infrastructure integration with Oracle Process Manager and Notification Server (OPMN), for process management, death detection, and failover for OC4J and Oracle HTTP Server processes.

> **See Also:**
>
> - *Oracle Application Server High Availability Guide*
> - *Oracle Process Manager and Notification Server Administrator's Guide*

Load Balancing

Oracle HTTP Server includes a module called `mod_oc4j` that routes requests from the OC4J instances in a cluster. OPMN helps ensure that `mod_oc4j` instances know of all the OC4J in the system without requiring a system administrator to do any configuration.

> **See Also:** Appendix A, "Load Balancing Using mod_oc4j"

Oracle HTTP Server Components

Oracle HTTP Server consists of several components that run within the same process. These components provide the extensive list of features that Oracle HTTP Server offers when handling client requests. The following are the major components:

- **HTTP Listener**: Oracle HTTP Server is based on an Apache HTTP listener to serve client requests. An HTTP server listener handles incoming requests and routes them to the appropriate processing utility.

- **Modules (mods)**: Modules both implement and extend the basic functionality of Oracle HTTP Server. Many of the standard Apache modules are included with Oracle HTTP Server. Oracle also includes several internal modules that are specific to Oracle Application Server components.

  See Also: "Oracle HTTP Server Modules" on page 1-4

- **Perl Interpreter**: A persistent Perl runtime environment embedded in Oracle HTTP Server through mod_perl.

  See Also: *Oracle Application Server Concepts*

Oracle HTTP Server Modules

Table 1–1 identifies the modules shipped with Oracle HTTP Server. Modules extend the basic functionality of the Web server, and support integration between Oracle HTTP Server and other Oracle Application Server components. Note that the list differs from the Apache open source distribution (given the inclusion of Oracle modules), and that not all modules are supported by Oracle.

Table 1–1 Oracle HTTP Server Modules

Module	Note	Module	Notes
mod_access		mod_log_config	
mod_actions		mod_logio	
mod_alias		mod_mime	
mod_asis		mod_mime_magic	
mod_auth		mod_negotiation	
mod_auth_anon		mod_oc4j	Oracle module.
mod_auth_dbm		mod_onsint	Oracle module.
mod_autoindex		mod_ossl	Oracle module.
mod_cern_meta		mod_osso	Oracle module.
mod_certheaders	Oracle module.	mod_perl	
mod_cgi		mod_php	
mod_cgid		mod_proxy	
mod_dir		mod_rewrite	
mod_env		mod_setenvif	
mod_expires		mod_speling	
mod_fastcgi		mod_status	
mod_file_cache		mod_unique_id	UNIX systems only.
mod_headers		mod_userdir	
mod_imap		mod_usertrack	
mod_include		mod_vhost_alias	
mod_info		mod_wchandshake	Oracle module.

See Also: Chapter 7, "Understanding Modules"

Oracle HTTP Server Support

Oracle provides technical support for the following Oracle HTTP Server features and conditions:

- Modules included in the Oracle distribution. Modules from any other source, including the Apache Software Foundation, are not supported by Oracle.
- Problems that can be reproduced within an Apache configuration consisting only of supported Oracle Apache modules.
- Use of the included Perl interpreter within the supported Apache configuration.

Oracle HTTP Server Management

You can manage Oracle HTTP Server using `opmnctl`. It is the command-line utility for Oracle Process Manager and Notification Server (OPMN) for process management. It is located in:

- UNIX: *`ORACLE_HOME`*`/opmn/bin`
- Windows: *`ORACLE_HOME`*`\opmn\bin`

> **See Also:** *Oracle Process Manager and Notification Server Administrator's Guide*

> **Note:** The Oracle HTTP Server 2.0 standalone install in Release 10.1.2 supports IPv6, but OPMN does not. If you want OPMN to manage, and successfully ping Oracle HTTP Server 2.0, you must have Oracle HTTP Server listen on at least one IPv4 address.

Starting, Stopping, and Restarting Oracle HTTP Server

Oracle HTTP Server is managed by Oracle Process Manager and Notification Server (OPMN). You must always use the `opmnctl` utility to start, stop, and restart the server. Otherwise, the configuration management infrastructure cannot detect or communicate with the Oracle HTTP Server processes, and problems may occur.

> **Note:** Do not use the `apachectl` utility to manage Oracle HTTP Server.

To determine the state of Oracle HTTP Server, use the following command:

```
opmnctl status
```

The processes are listed with their current state (Up, Down, and so on)

Starting Oracle HTTP Server

To start Oracle HTTP Server, use the `startproc` command:

- UNIX: *`ORACLE_HOME`*`/opmn/bin> opmnctl [verbose] startproc ias-component=HTTP_Server`
- Windows: *`ORACLE_HOME`*`\opmn\bin> opmnctl [verbose] startproc ias-component=HTTP_Server`

Stopping Oracle HTTP Server

To stop Oracle HTTP Server, use the `stopproc` command:

- UNIX: `ORACLE_HOME/opmn/bin> opmnctl [verbose] stopproc ias-component=HTTP_Server`
- Windows: `ORACLE_HOME\opmn\bin> opmnctl [verbose] stopproc ias-component=HTTP_Server`

Restarting Oracle HTTP Server

Restarting Oracle HTTP Server performs a graceful restart, which is invisible to clients. In a graceful restart, on UNIX, a `USR1` signal is sent. When the process receives this signal, it tells the children to exit after processing the current request. (Children that are not servicing requests exit immediately.)

The parent re-reads the configuration files and re-opens the log files, replacing the children with new children in accordance with the settings it finds when re-reading the configuration files. It always observes the process creation settings (`MaxClients`, `MaxSpareServers`, `MinSpareServers`) specified, and takes the current server load into account.

To restart Oracle HTTP Server, use the `restartproc` command:

- UNIX: `ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`
- Windows: `ORACLE_HOME\opmn\bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`

> **See Also:** *Oracle Process Manager and Notification Server Administrator's Guide*

2

Configuring Standalone Oracle HTTP Server with Oracle Application Server

This chapter contains information about configuring a standalone Oracle HTTP Server 2.0 installation to communicate with an existing Oracle Application Server, 10*g* Release 2 (10.1.2) middle-tier installation. The standalone installation of Oracle Application Server for Oracle HTTP Server 2.0 does not contain Oracle Enterprise Manager 10*g* Grid Control Console, and Distributed Configuration Management, which make configuration and management of groups of servers (farms) simple in the Oracle Application Server installation. Without these helpful tools, several manual configuration steps are needed to configure the standalone Oracle HTTP Server 2.0 installation to interoperate with an existing managed Oracle Application Server middle-tier installation.

Topics discussed are:

- Configuration Checklist
- Installing Standalone Oracle HTTP Server
- Configuring OPMN
- Configuring mod_oc4j
- Configuring Single Sign-On

Configuration Checklist

Before configuring the standalone Oracle HTTP Server 2.0 installation, verify the following:

- Assure that all the standard (managed) Oracle Application Server, 10*g* Release 2 (10.1.2), instances are installed and configured as desired. You must configure all instances prior to configuration of the standalone Oracle HTTP Server 2.0 listener. Any changes to the standard Oracle Application Server OC4J configuration, such as adding and removing a server from a cluster, or adding a new instance, will require a reconfiguration of the standalone (manually managed) installation.

  See Also:

  Oracle Application Server Administrator's Guide

  Oracle Application Server High Availability Guide

- Install and configure the standalone Oracle HTTP Server 2.0. These instructions assume that all instances of the Oracle HTTP Server 2.0 and the regular Oracle

Application Server middle-tier have been installed prior to proceeding with the configuration steps described in the following sections.

Installing Standalone Oracle HTTP Server

Standalone Oracle HTTP Server is distributed on the OracleAS Companion CD, which is included in the Oracle Application Server CD Pack.

Following are instructions for installing standalone Oracle HTTP Server:

1. Insert the OracleAS Companion CD, and do the following to launch the Oracle Universal Installer to install standalone Oracle HTTP Server:

   - On UNIX:

     ```
     prompt > cd
     ```

     ```
     prompt > mount_point/1012disk1/runInstaller
     ```

   - On Windows:

     If your computer supports the auto-run feature, then the installer launches automatically.

     If your computer does not support auto-run, then double click on the `setup.exe` file to launch the installer.

2. When Oracle Universal Installer appears, review the Welcome screen, and click **Next**.

3. If this is the first time you are installing any Oracle products on your computer, the Specify Inventory Directory and Credentials screen appears:

   Enter the following information on this screen:

   – Full Path of the Inventory Directory: Enter the full path to a directory for the installer's files. Enter a directory that is different from the Oracle home directory for the product files.

     For example: `/opt/oracle/oraInventory`

   – Operating System Group Name: Enter the name of the operating system group that has write permissions for the inventory directory.

     For example: `oinstall`

   Click **Next**. A window appears and asks you to run `orainstRoot.sh`. Run the script in a different shell as the `root` user. The script is located in the `oraInventory` directory. Click **Continue**.

4. On the Specify File Locations screen, enter the following information:

   - **Name**: Enter a name to identity this Oracle home. The name cannot contain spaces, and has a maximum of 16 characters.

     For example: `OH_STANDOHS`

   - **Destination Path**: Enter the full path to the destination directory. This is the Oracle home. If the directory does not exist, the installer creates it. If you want to create the directory beforehand, create it as the `oracle` user; do not create it as the `root` user.

     For example: `/opt/oracle/STANDOHS`

   Click **Next**.

5. On the Select a Product to Install screen, select Web Server Services 10.1.2.0.0, and click **Next**.
6. On the Select Installation Type screen, select the standalone Oracle HTTP Server installation of your choice, and click **Next**.
7. On the Summary screen, verify your selections, and click **Install**.
8. The Install Progress screen displays the progress of the installation.
9. On the Configuration Assistants screen, monitor the progress of the configuration assistants. The configuration assistants configure the installed components. You will be prompted to run `root.sh`. Run the script in a different shell as the `root` user. Click **OK**.
10. The End of Installation screen appears once the installation has completed. Click **Exit** to quit the installer.

See Also: *Oracle Application Server Installation Guide* for more information on Oracle Universal Installer.

Configuring OPMN

Oracle Process Manager and Notification Server (OPMN) consists of the following two components that interpret and convey notification sent between Oracle Application Server processes within the same or different OPMN servers:

- **Oracle Notification Server**: Oracle Notification Server (ONS) is the transport mechanism for failure, recovery, startup, and other related notifications between components in Oracle Application Server. It operates according to a publish-subscribe model: an Oracle Application Server component receives a notification of a certain type per its subscription to ONS. When such a notification is published, ONS sends it to the appropriate subscribers.
- **Oracle Process Manager**: Oracle Process Manager (PM) is the centralized process management mechanism in Oracle Application Server and is used to manage Oracle Application Server processes. It starts, stops, restarts, and detects death of these processes. The Oracle Application Server processes that PM is configured to manage are specified in the `opmn.xml` file.

See Also: *Oracle Process Manager and Notification Server Administrator's Guide*

Perform the following steps to configure Oracle Process Manager and Notification Server.

1. Copy the `ons.conf` configuration file from the regular Oracle Application Server middle-tier installation to the corresponding directory in the Oracle HTTP Server 2.0 installation. This file is located in the *`ORACLE_HOME`*`/opmn/conf` directory. Edit the `ons.conf` file to add all of the Oracle HTTP Server 2.0 standalone instances in the manually managed cluster. The new `ons.conf` file should contain a list of all the instances in the managed Oracle Application Server installation, and each of the manually managed instances as well.

   The following is the `ons.conf` file format:

   ```
   nodes=<host_name | host_ip>[:port] [,<host_name | host_ip>[:port]] [, ...]
   ```

   For example:

   ```
   nodes=managed1:6000,managed1:6300,unmanaged2:6400
   ```

2. To determine the correct ONS remote listening port, examine the OPMN configuration file of each manually managed instance (located at *ORACLE_HOME*/opmn/conf/opmn.xml). The ONS remote listening port value is specified by the "remote" attribute of XML element located at /opmn/notification-server/port in the opmn.xml file.

   For example:

   If opmn.xml on unmanaged2 contains:

   ```
   <opmn>
     <notification-server>
       <port local='6100' remote='6400' request='6300'/>
     ...
     </notification-server>
   <...>
   </opmn>
   ```

   Then, the ons.conf file should contain:

   ```
   nodes=managed1:6200,unmanaged1:6300,unmanaged2:6400
   ```

3. If a host is multi-homed (is configured with multiple IP addresses), it is best to set the /opmn/notification-server/ipaddr "remote" attribute in the opmn.xml file. This attribute will bind the ONS listener to a single valid IPv4 address or host name. If this attribute is not set, or the ipaddr element is omitted, then ONS will enable listening on all IP addresses on a multi-homed host.

   For example:

   The host unmanaged2 is multi-homed, with IP addresses of 10.1.1.1 and 10.1.2.1. To restrict ONS to listen on only the 10.1.1.1 IP address, modify the opmn.xml file as follows:

   ```
   <opmn>
     <notification-server>
       <ipaddr remote='10.1.1.1'/>
       <port local='6100' remote='6400' request='6300'/>
     ...
     </notification-server>
   <...>
   </opmn>
   ```

   ons.conf should look like the following:

   ```
   nodes-managed1.oracle.com:6200,unmanaged1.oracle.com:6300,10.1.1.1.1:6400
   ```

4. Make sure to create an entry in the ons.conf file for every single Oracle Application Server instance in the cluster. Copy this manually created file to each of the other unmanaged Oracle HTTP Server 2.0 instances in the cluster. The ons.conf configuration data should match both the "remote" and "ipaddr" setting, if present, in the opmn.xml configuration file for each unmanaged node.

Configuring mod_oc4j

Manually managed Oracle HTTP Server 2.0 listeners must be configured to route traffic to the managed Oracle Application Server Containers for J2EE (OC4J) installations as follows:

1. Configure OC4J on the managed cluster.

See Also:

- *Oracle HTTP Server Administrator's Guide*
- *Oracle Application Server Containers for J2EE User's Guide*

2. For each manually managed standalone Oracle HTTP Server 2.0 instance, assure that the `mod_oc4j.conf` file is configured to point to the managed cluster and instance.

   For example, a manually managed Oracle HTTP Server 2.0 listener is configured to direct traffic to the cluster named `managed1`, instance name `home` is used in the `mod_oc4j.conf` file:

   ```
   Oc4jMount /MyApp/* cluster://managed1:home
   ```

3. A mount point must be added for each application for which routing is needed.
4. Restart Oracle HTTP Server to allow configuration changes to take effect.
   - UNIX: `ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`
   - Windows: `ORACLE_HOME\opmn\bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`
5. Each time a new application is configured, Oracle HTTP Server 2.0 standalone `mod_oc4j.conf` must be modified to reflect these changes.

Configuring Single Sign-On

If single sign-on functionality is desired for the standalone Oracle HTTP Server 2.0 install, manually managed Oracle HTTP Server 2.0 listeners may be registered with Oracle Application Server Single Sign-On by performing the following steps:

1. Configure the Partner Application using SSO Server Administration tool.

   See Also: *Oracle Application Server Single Sign-On Administrator's Guide*

2. Manually create the `osso.conf` file. This is accomplished by cutting and pasting the required data from the Edit Partner Application screen, after the Partner Application had been configured.

   For example, the Edit Partner Application screen will display the necessary configuration data for creating the `osso.conf` file. The following shows an Edit Partner Application page sample:

   ```
   ID: 643C32F6
   Token: Q2057R2D646C20F1
   Encryption Key: 3F46C27C5153B7C7
   Login URL: http://foobar.us.oracle.com:7778/pls/orasso.wwsso_app_admin.ls_login
   Single Sign-Off: http://foobar.us.oracle.com:7778/pls/orasso.wwsso_app_
   admin.ls_logout
   ```

   The data provided from the Edit Partner Application screen can be used to manually create a clear-text `osso.conf` configuration file as follows:

   ```
   sso_server_version=v1.4
   cipher_key=3F46C27C5153B7C7
   site_id=643C32F6
   site_token=Q2057R2D646C20F1
   ```

```
login_url=http://foobar.us.oracle.com:7778/pls/orasso.wwsso_app_admin.ls_login
logout_url=http://foobar.us.oracle.com:7778/pls/orasso.wwsso_app_admin.ls_
logout
cancel_url=http://foobar.us.oracle.com:7778
```

3. Copy the newly created file to the `osso` configuration directory:

   ```
   ORACLE_HOME/ohs/conf/osso.conf cleartext
   ```

4. The plain-text file must now be obfuscated to protect the encryption key information. This is accomplished by using the `apobfuscate` tool located in *`ORACLE_HOME`*`/Apache/Apache/bin` directory as following:

   ```
   ../../bin/apobfuscate osso/conf/cleartext osso.conf
   ```

5. Edit the standalone Oracle HTTP Server 2.0 `mod_osso.conf` to enable SSO Web resource protection. This file is located in the *`ORACLE_HOME`*`/ohs/conf` directory. Make sure the `OssoConfigFile` directive points to the obfuscated `osso.conf` file containing the Partner Application registration data. Also, assure that the `#include "`*`ORACLE_HOME`*`/ohs/conf/mod_osso.conf"` directive is uncommented in the `httpd.conf` file.

   For example:

   ```
   LoadModule osso_module libexec/mod_osso.so
   <IfModule mod_osso.c>
   OssoConfigFile conf/osso/osso.conf
   OssoIpCheck off
   OssoIdleTimeout off

   Alias /private/ "<ApacheServerRoot>/private/"

   <Location /private>
   require valid_user
   AuthType Basic
   </Location>

   </IfModule>
   ```

6. Restart Oracle HTTP Server to allow the configuration to take effect.

   - UNIX: *`ORACLE_HOME`*`/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`
   - Windows: *`ORACLE_HOME`*`\opmn\bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`

3

Specifying Server and File Locations

This chapter explains how to set Oracle HTTP Server and server administrator options, and specifies file locations.

Topics discussed are:

- Setting Server and Administrator Functions
- Specifying File Locations

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Setting Server and Administrator Functions

The following directives set basic Oracle HTTP Server and administrator functions. They are located in the "Main Server Configuration" portion of the `httpd.conf` file.

> **See Also:** "httpd.conf File Structure" on page B-2

- ServerName
- UseCanonicalName
- ServerAdmin
- ServerSignature
- ServerTokens
- ServerAlias

ServerName

Enables the server to set a hostname that can be used to create redirection URLs, through which you can access directories without having to use a "/" at the end.

For example, `ServerName www.company.com` would be used if the main name of the actual machine were `main.company.com`.

> **See Also:** "`ServerName` directive" in the Apache Server documentation.

UseCanonicalName

Determines which hostname and port to use when redirecting the URL to the same server.

- `On`: Server uses the hostname and port values set in ServerName. This is the default setting.
- `Off`: Server uses the hostname and port that you specify in the request.

  For example: `UseCanonicalName On`.

> **See Also:** "`UseCanonicalName` directive" in the Apache Server documentation.

ServerAdmin

Creates an email address that is included with every default error message that clients encounter. It is useful to create a separate email address for this.

For example: `ServerAdmin you@your.emailaddress`.

> **See Also:** "`ServerAdmin` directive" in the Apache Server documentation.

ServerSignature

Enables the server to recognize which server, among the various proxies, created the returned response, such as an error message.

- `on`: Server creates a footer to the returned document that includes information such as ServerName and server version number. This is the default setting.
- `email`: Server creates an additional "mailto:" reference to the ServerAdmin of the document.
- `off`: Footer and "mailto:" reference are not created.

For example: `ServerSignature On`

> **See Also:** "`ServerSignature` directive" in the Apache Server documentation.

ServerTokens

Controls server information which is returned to clients, such as in error messages. This information includes a description of the generic operating system-type of the server, and compiled-in modules.

- `min(imal)`: provides information such as server name and version.
- `OS`: provides information such as server name, version and operating system.
- `full`: provides information such as server name, version, operating system, and complied modules.

For example: `Server: Apache/2.0.0 (UNIX) PHP/3.0 MyMod/1.2`

> **See Also:** "`ServerTokens` directive" in the Apache Server documentation.

ServerAlias

Sets alternate names for the current virtual host.

For example:

```
<VirtualHost *>
ServerName server.domain.com
ServerAlias server server2.domain.com server2
...
</VirtualHost>
```

> **See Also:** "`ServerAlias` directive" in the Apache Server documentation.

Specifying File Locations

The following directives control the location of various server files. They are located in the "Global Environment" of the `httpd.conf` file.

> **See Also:** "httpd.conf File Structure" on page B-2

- CoreDumpDirectory
- DocumentRoot
- ErrorLog
- LockFile
- PidFile
- ScoreBoardFile
- ServerRoot

CoreDumpDirectory

Specifies the directory in which the server dumps core. The default is the ServerRoot directory. This directive is applicable to UNIX only.

For example: `CoreDumpDirectory /tmp`

> **See Also:** "`CoreDumpDirectory` directive" in the Apache Server documentation.

DocumentRoot

Sets the directory from which httpd serves files. Unless matched by a directive like `Alias`, the server appends the path from the requested URL to the document root to make the path to the document for static content.

For example: `DocumentRoot "/oracle/Apache/Apache/htdocs"`

> **See Also:** "`DocumentRoot` directive" in the Apache Server documentation.

ErrorLog

Sets the name of the file to which the server notes any errors it encounters. If the name of the file does not begin with a slash (/), then it is assumed to be relative to the

ServerRoot. If the name of the file begins with a pipe (|), then it is assumed to be a command to spawn to handle the error log.

For example: `ErrorLog "|/private1/oracle/Apache/Apache/bin/rotatelogs /private1/oracle/Apache/Apache/logs/error_log 43200"`

> **See Also:** "`ErrorLog` directive" in the Apache Server documentation.

LockFile

Sets the path to the lockfile used whenOracle HTTP Server is compiled with either `USE_FCNTL_SERIALIZED_ACCEPT` or `USE_FLOCK_SERIALIZED_ACCEPT`. It is recommended that default value be used. The main reason for changing it is if the logs directory is NFS mounted, since the lockfile must be stored on a local disk.

For example: `LockFile /oracle/Apache/Apache/logs/httpd.lock`

> **See Also:** "`LockFile` directive" in the Apache Server documentation.

PidFile

Enables you to set and change the location of the `PID` file to which the server records the process identification number. If the filename does not begin with a slash (/), then it is assumed to be relative to the ServerRoot.

For example: `PidFile /oracle/Apache/Apache/logs/httpd.lock`

> **See Also:** "`PidFile` directive" in the Apache Server documentation.

ScoreBoardFile

Required in some architectures to set a file that the server uses to communicate between the parent and children processes. To verify if your architecture requires a scoreboard file, run Oracle HTTP Server and see if it creates the file named by the directive. If your architecture requires it then you must ensure that this file is not used at the same time by more than one invocation of the server.

For example: `/oracle/Apache/Apache/logs/httpd.scoreboard`

> **See Also:** "`ScoreBoardFile` directive" in the Apache Server documentation.

ServerRoot

Specifies the directory that contains the `conf` and `logs` subdirectories. If the server is started with the `-f` option, then you will have to specify ServerRoot.

For example: `"/oracle/Apache/Apache"`

> **See Also:** "`ServerRoot` directive" in the Apache Server documentation.

4 Managing Server Processes

This chapter provides an overview of the Oracle HTTP Server processes, and provides information on how to regulate, and monitor these processes.

Topics discussed are:

- Oracle HTTP Server Processing Model
- Handling Server Processes
- Configuring the Number of Processes and Connections
- Running Oracle HTTP Server as Root
- Security Considerations
- Getting Information about Processes

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Oracle HTTP Server Processing Model

Once Oracle HTTP Server is started, the system is ready to listen for and respond to http(s) requests. The request processing model on UNIX differs from that on Windows.

On UNIX, there is a single parent process that manages multiple child processes. The child processes are responsible for handling requests. The parent process brings up additional child processes as necessary, based on configuration. Although the server has the ability to dynamically bring up additional child processes, it is best to configure the server to start enough children initially so that requests can be handled without having to spawn more child processes.

On Windows, there is a single parent process and a single child process. The child process creates threads that are responsible for handling client requests. The number of threads created is static and can be configured.

Handling Server Processes

By default, on UNIX, the main httpd parent process and child processes are configured to run as the user who installed Oracle Application Server. The User and Group directives are used to set the privileges for the child processes. These directives are ignored if you are not running as `root`. The child processes must be able to read all the content that will be served.

- Group
- User

Group

Specifies the group under which the server answers requests. Run the standalone server as `root` to use this directive. It is recommended that you create a new group for running the server. This is applicable to UNIX only.

For example: `Group myorg`

> **See Also:** "`Group` directive" in the Apache Server documentation.

User

Specifies the user ID to which the server answers requests. Run the standalone server as `root` to use this directive.You should have privileges to access files that are available for everyone, and should not be able to execute code which is not meant for httpd requests. It is recommended that you set up a new user for running the server. This is applicable to UNIX only.

For example: `User jdoe`

> **See Also:** "`User` directive" in the Apache Server documentation.

Configuring the Number of Processes and Connections

The following directives tune the performance of Oracle HTTP Server by configuring how clients requests are processed. They are located in the "Global Environment" of the `httpd.conf` file.

> **See Also:** "httpd.conf File Structure" on page B-2

- StartServers
- ThreadsPerChild
- MaxClients
- MaxRequestsPerChild
- MaxSpareServers
- MinSpareServers

StartServers

Sets the number of child server processes created when Oracle HTTP Server is started. The default is 5. This is applicable to UNIX only.

Usage: `StartServers 5`

> **See Also:** "`StartServers` directive" in the Apache Server documentation.

ThreadsPerChild

Controls the maximum number of child threads handling requests. The default is 50. This is applicable to Windows only.

Usage: `ThreadsPerChild 50`

See Also: "`ThreadsPerChild` directive" in the Apache Server documentation.

MaxClients

Limits the number of requests that can be dealt with at one time. The default and recommended value is 150. This is applicable to UNIX only.

Usage: `MaxClients 150`

See Also: "`MaxClients` directive" in the Apache Server documentation.

MaxRequestsPerChild

Controls the number of requests a child process handles before it dies. If you set the value to 0, which is the default, then the process will never die.

On Windows, it is recommended that this be set to 0. If it is set to a non-zero value, when the request count is reached, the child process exits, and is respawned, at which time it re-reads the configuration file. This can lead to unexpected behavior if you have modified a configuration file, but are not expecting the changes to be applied yet.

Usage: `MaxRequestsPerChild 0`

See Also: "`MaxRequestsPerChild` directive" in the Apache Server documentation.

MaxSpareServers

Sets the maximum number of idle child server processes. An idle process is one which is running, but not handling a request. The parent process kills off idle child processes that exceed the value set for this directive. The default is 20. This is applicable to UNIX only.

Usage: `MaxSpareServers 20`

See Also: "`MaxSpareServers` directive" in the Apache Server documentation.

MinSpareServers

Sets the minimum number of idle child server processes. An idle process is one which is running but not handling a request. The parent process will create new children at the maximum rate of one process per second if there are fewer processes running. The default is 5. This is applicable to UNIX only.

Usage: `MinSpareServers 5`

See Also: "`MinSpareServers` directive" in the Apache Server documentation.

Running Oracle HTTP Server as Root

On UNIX, if you want to run on ports less than 1024, then you will have to run as `root`.

In order to run Oracle HTTP Server as `root`, perform the following steps:

1. Stop Oracle HTTP Server using the following command:

```
ORACLE_HOME/opmn/bin> opmnctl [verbose] stopproc ias-component=HTTP_Server
```

2. Change to root user.
3. Navigate to *`ORACLE_HOME`*`/ohs/bin` and execute the following command:

```
chown root .apachectl
chmod 6750 .apachectl
```

4. Exit `root`.
5. Restart Oracle HTTP Server using the following command:

```
ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server
```

Security Considerations

For additional security on UNIX, you can change the user to "nobody". Be sure that the child processes can accomplish their tasks as the user "nobody". Change all static content, such as the *`ORACLE_HOME`*`/ohs/htdocs` directory on UNIX or *`ORACLE_HOME`*`\ohs\htdocs` on Windows, so that all the files are readable, but ideally not writable by the user "nobody". Also, verify that all the CGI and FastCGI programs can be run by user "nobody".

Finally, given that the cached content might contain sensitive data, the final contents of the file-system cache should be protected. So, although Oracle HTTP Server might run as "nobody", access to the system as this user should be well-protected.

Getting Information about Processes

There are several ways to monitor Oracle HTTP Server processes.

1. Use the performance monitor on Windows, or the `ps` utility on UNIX.

   See Also: *Oracle Application Server Performance Guide* and your operating system documentation for more information.

2. Use mod_status for server status. By default, it is available from localhost only.

5

Managing the Network Connections

This chapter provides information about specifying IP addresses and ports, and managing server interaction, and network connection persistence.

Topics discussed are:

- Specifying Listener Ports and Addresses
- Managing Interaction Between Server and Network
- Managing Connection Persistence
- Obtaining Client IP Address
- Configuring Reverse Proxies and Load Balancers

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Specifying Listener Ports and Addresses

The port that Oracle HTTP Server listens on when it is started depends on your installation type.

Table 5–1 contains information about Oracle HTTP Server ports.

Table 5–1 Oracle HTTP Server Ports

Platform	Middle Tier Installation	Infrastructure Installation
Solaris	Non-SSL: 7777 (7777-7877 range) SSL: 4443 (4443-4543 range)	Non-SSL: 7777 (7777-7877 range) SSL: 4443 (4443-4543 range)
Windows	Non-SSL: 80 (7777-7877 range) SSL: 443 (4443-4543 range)	Non-SSL: 7777 (7777-7877 range) SSL: 4443 (4443-4543 range)

If ports 7777 or 80, for example, are occupied, Oracle HTTP Server listens on the next available port numbers between a range of 7777-7877. Accordingly, it would listen on port 7778, and so on.

Note: SSL is disabled by default. For information on enabling SSL, refer to Chapter 9, "Enabling SSL for Oracle HTTP Server".

A file named `setupinfo.txt` is automatically generated in *`ORACLE_HOME`*`/install` on UNIX or *`ORACLE_HOME`*`\install` on Windows. It contains port information for Oracle HTTP Server. This file is generated at install time, and is not updated thereafter. If you restart Oracle HTTP Server, the information in this file becomes inaccurate.

You can change the Oracle HTTP Server listener port (SSL and non-SSL) after installation. If you make a port change, then you have to also update other components to use the new port number.

See Also: *Oracle Application Server Administrator's Guide*

You can specify that the server listens on more than one port, selected addresses, or a combination. The Listen directive, located in the "Global Environment" of the httpd.conf file, specifies listener ports and addresses. Multiple `Listen` directives can be used to listen on multiple ports.

For example:

- `Listen 7778`
- `Listen 12.34.56.78:80`

See Also: "`Listen` directive" in the Apache Server documentation.

Managing Interaction Between Server and Network

The following directives are used to specify how the server interacts with the network. They are located in the "Global Environment" of the `httpd.conf` file.

- ListenBackLog
- SendBufferSize
- TimeOut

See Also: "httpd.conf File Structure" on page B-2

ListenBackLog

Specifies the maximum length of the queue of pending connections. This is useful if the server is experiencing a `TCP SYN` overload, which causes numerous new connections that open up, but do not complete the task.

See Also: "`ListenBackLog` directive" in the Apache Server documentation.

SendBufferSize

Increases the `TCP` buffer size to the number of bytes specified, thereby improving performance.

See Also: "`SendBufferSize` directive" in the Apache Server documentation.

TimeOut

Sets the maximum time, in seconds, that the server waits for the following:

- The total amount of time it takes to receive a `GET` request.
- The amount of time between receipt of `TCP` packets on a `POST` or `PUT` request.
- The amount of time between `ACKs` on transmissions of `TCP` packets in responses.

The default is 300 seconds.

> **See Also:** "`TimeOut` directive" in the Apache Server documentation.

Managing Connection Persistence

The following directives determine how the server handles persistent connections. They are located in the "Global Environment" of the httpd.conf file.

- KeepAlive
- KeepAliveTimeout
- MaxKeepAliveRequests

> **See Also:**
>
> - *Oracle Application Server Performance Guide*
> - "httpd.conf File Structure" on page B-2

KeepAlive

Enables HTTP 1.1 keep-alive support, allowing reuse of the same TCP connection for multiple HTTP requests from a single client, when set to "On".

> **See Also:** "`KeepAlive` directive" in the Apache Server documentation.

KeepAliveTimeout

Sets the number of seconds the server waits for a subsequent request before closing a KeepAlive connection. Once a request has been received, the timeout value specified by the TimeOut directive applies. The default is 15 seconds.

> **See Also:** "`KeepAliveTimeout` directive" in the Apache Server documentation.

MaxKeepAliveRequests

Limits the number of requests allowed per connection when KeepAlive is on. If it is set to "0", unlimited requests will be allowed. The default is 100.

> **See Also:** "`MaxKeepAliveRequests` directive" in the Apache Server documentation.

Obtaining Client IP Address

UseWebCacheIp is a global directive that enables Oracle HTTP Server to obtain IP address of a client. It can be set to "On" or "Off", and defaults to "Off". It is not set to "On" by default because it can open a security hole in some circumstances.

When OracleAS Web Cache acts as a reverse proxy in front of Oracle HTTP Server, the TCP connection from the client is terminated at OracleAS Web Cache. The TCP connection that Oracle HTTP Server sees actually originates at OracleAS Web Cache. Oracle HTTP Server gets the IP address of the client and uses it for various purposes, such as:

- Populating the REMOTE_ADDR CGI variable that can be used by applications in and behind Oracle HTTP Server to identify where the client came from.
- Evaluating mod_access allow/deny rules that allow the administrator to restrict access based on IP address.

Without the UseWebCacheIp directive, this functionality fails when OracleAS Web Cache is used in front of Oracle HTTP Server. This is because Oracle HTTP Server sees all connections coming from the same place, the IP address where OracleAS Web Cache is running.

With every request that OracleAS Web Cache forwards to Oracle HTTP Server, it sends a header that contains the IP address of the client connection that it received. If UseWebCacheIp is set to "On", then it directs Oracle HTTP Server to use the IP value from this header, instead of the value from the TCP connection as the client's IP address. This enables REMOTE_ADDR CGI variable to have the correct value, and allows mod_access to function correctly.

You should set this directive only if you are sure that the clients can only connect to Oracle HTTP Server through OracleAS Web Cache. If clients can connect directly to Oracle HTTP Server, then they have to find out the header that is used to transfer the client IP, and set it so that it would seem to have come from any IP address you want. In a typical set up, with a firewall and OracleAS Web Cache, the only port open through the firewall is the OracleAS Web Cache port. Hence, the only path from the client to Oracle HTTP Server goes through OracleAS Web Cache. In this case, it is safe to turn on UseWebCacheIp.

Configuring Reverse Proxies and Load Balancers

By default, Oracle Application Server installs using the local hostname as set up by ServerName directive in Oracle HTTP Server. Most Web sites tend to have a specific hostname or domain name for their Web or application server. However, this is not possible out of the box because with the ServerName directive, Oracle HTTP Server is instantiated with the local host.

Example 5–1 Using Reverse Proxies and Load Balancers with Oracle HTTP Server

Domain Name: www.oracle.com:80 123.456.7.8 (hosted on a reverse proxy, load balancer, or firewall)

Host Name of Oracle Application Server Host: server.oracle.com 123.456.7.9

ServerName and Port of Oracle Application Server Host: server.oracle.com:7777

Make the following changes in the httpd.conf file:

```
Port 80
```

```
Listen 7777
Listen 80
# Virtual Hosts
# This section is mandatory for URLs that are generated by
# the PL/SQL packages of the Oracle Portal and various other components
# These entries dictate that the server should listen on port
# 7777, but will assert that it is using port 80, so that
# self-referential URLs generated specify www.oracle.com:80
# This will create URLs that are valid for the browser since
# the browser does not directly see the host server.oracle.com.
NameVirtualHost 123.456.7.9:7777
<VirtualHost server.oracle.com:7777>
ServerName www.oracle.com
Port 80
</VirtualHost>
# Since the previous virtual host entry will cause all links
# generated by the Oracle Portal to use port 80, the server.company.com
# server needs to listen on 80 as well since the Parallel Page
# Engine will make connection requests to Port 80 to request the
# portlets.
NameVirtualHost 123.456.7.9:80
<VirtualHost server.oracle.com:80>
ServerName www.oracle.com
Port 80
<VirtualHost>
```

See Also: *Oracle Application Server High Availability Guide*

6
Configuring and Using Server Logs

This chapter discusses Oracle Diagnostic Logging, log formats, and describes various log files and their locations.

Topics discussed are:

- Using Oracle Diagnostic Logging
- Specifying Log Level
- Specifying Log Files

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Using Oracle Diagnostic Logging

Oracle offers a new method for reporting diagnostic messages. This new method, Oracle Diagnostic Logging (ODL), presents a common format for diagnostic messages and log files, and a mechanism for correlating all diagnostic messages from various components across Oracle Application Server. Using ODL, each component logs messages to its own private local repository. A tool called `LogLoader` collects messages from each repository and loads them into a common repository where messages can be viewed as a single log stream, or analyzed in different ways.

You can view Oracle Application Server diagnostic log files using a text editor.

> **See Also:** *Oracle Application Server Administrator's Guide*

ODL is further discussed in the following sections:

- Overview
- Configuring Oracle HTTP Server

Overview

Oracle HTTP Server enables you to choose the format in which you want to generate log messages. You can either continue to generate log messages in the legacy Apache message format, or generate log messages using ODL, which complies with the new Oracle-wide standards for generating log messages.

Configuring Oracle HTTP Server

To enable Oracle HTTP Server to use ODL, enter the following directives in the `httpd.conf` file:

- OraLogMode oracle | odl | apache
- OraLogSeverity module_name <msg_type>{:msg_level]
- OraLogDir <bus stop dir>

Oracle recommends that you enter the directives before any modules are loaded (`LoadModule` directive) in the `httpd.conf` file so that module-specific logging severities are in effect before modules have the opportunity to perform any logging.

OraLogMode oracle | odl | apache

Enables you to switch between the Oracle logging format, the legacy Apache logging format, and the ODL logging format. Logging formats are defined as follows:

- **oracle**: Fully conformant, multi-line log records in XML format. Provides the most information.
- **odl**: Standard Apache log format and ECID information for log records specifically associated with a request. This is the default setting.
- **apache**: Standard Apache log format. Provides the least information.

OraLogSeverity module_name <msg_type>{:msg_level]

Enables you to set message severity. The message severity specified with this directives is interpreted as the lowest message severity that is desired, and all messages of that severity level and higher will be logged.

`OraLogSeverity` may be specified multiple times. It can be specified globally (no module_name) and once for each module for which a module-specific logging severity is desired.

This directive is only used when `OraLogMode` is set to "oracle". This directive can be used in place of the `LogLevel` directive, but is not required. If `OraLogSeverity` is present and `OraLogMode` is set to "oracle", then `LogLevel` will be ignored.

module_name This argument is the internal name of a module, as it appears in the module structure. The `<IfModule>` directive also makes use of this internal name. The module structure derives the module name from the value of the `_FILE_` macro, without path prefix, of the file which defines the module structure. If a module name is not supplied, the `OraLogSeverity` directive is applied globally.

If the module name is specified, then the directive overrides the global directive value of all the messages originating from the specified module. Specifying a module name for a module that does not get loaded generates an error.

msg_type Message types may be specified in upper or lower case, but will appear in the message output in upper case. This parameter must be of one of the following values:

- `INTERNAL_ERROR`
- `ERROR`
- `WARNING`
- `NOTIFICATION`

- TRACE

msg_level This parameter must be an integer in the range of 1-32. 1 is most severe, 32 is least severe. Using level 1 will result in fewer messages than using level 32.

Table 6–1 lists some examples of `OraLogSeverity`.

Table 6–1 Examples of OraLogSeverity

OraLogSeverity Example	Action Taken
`OraLogSeverity INTERNAL_ERROR:10`	Logs all messages of type "internal error" of levels 1-10
`OraLogSeverity WARNING:7`	Logs all messages of type "internal error" of all levels Logs all messages of type "error" of all levels Logs all messages of type "warning" of levels 1-7

`OraLogSeverity WARNING`
`OraLogSeverity mod_oc4j.c NOTIFICATION:4`
	If message source is `mod_oc4j`, then
	■ Logs all messages of type "internal error" of all levels
	■ Logs all messages of type "error" of all levels
	■ Logs all messages of type "warning" of all levels
	■ Logs all messages of type "notification" of levels 1-4
	For messages from all other sources:
	■ Logs all messages of type "internal error" of all levels
	■ Logs all messages of type "error" of all levels
	■ Logs all messages of type "warning" of all levels

Default If a message level is not specified, then the level defaults to the lowest severity. If the entire directive is omitted, then the value of the global Apache `LogLevel` directive is used and translated to the corresponding Oracle message type and the lowest level within the corresponding range, as listed in Table 6–2:

Table 6–2 Apache Log Level and Corresponding Oracle Message Type

Apache Log Level	Oracle Message Type
`emerg`	`INTERNAL_ERROR:16`
`alert`	`INTERNAL_ERROR:32`
`crit`	`ERROR:16`
`error`	`ERROR:32`
`warn`	`WARNING:32`
`notice`	`NOTIFICATION:16`
`info`	`NOTIFICATION:32`
`debug`	`TRACE:32`

> **See Also:** "Specifying Log Level" on page 6-4

OraLogDir <bus stop dir>

Specifies the path to the directory which contains all log files. This directory must exist.

Default:

- UNIX: *ORACLE_HOME*/Apache/Apache/logs/oracle
- Windows: *ORACLE_HOME*\Apache\Apache\logs\oracle

Specifying Log Level

Table 6–3 lists all the different logging levels, their descriptions, and, example messages:

Table 6–3 Logging Level

Logging Level	Description	Example Message
emerg	Emergencies- system is unusable.	"Child cannot open lock file. Exiting."
alert	Action must be taken immediately.	"getpwuid: couldn't determine user name from uid"
crit	Critical conditions.	"socket: Failed to get a socket, exiting child"
error	Error conditions.	"Premature end of script headers"
warn	Warning conditions.	"child process 1234 did not exit, sending another SIGHUP"
notice	Normal but significant condition.	"httpd: caught SIGBUS, attempting to dump core in..."
info	Informational.	"Server seems busy, (you may need to increase StartServers, or Min/MaxSpareServers)..."
debug	Debug-level messages.	"Opening config file..."

Note: LogLevel directive may be omitted when OraLogMode is "oracle' and OraLogSeverity is set.

Specifying Log Files

The following section describes the function and location of these log files:

- Access Log
- CustomLog
- Error Log
- PID File
- Piped Log
- Rewrite Log
- Script Log
- SSL Log
- Transfer Log

It is important to periodically rotate the log files by moving or deleting existing logs on a moderately busy server. For this, the server must be restarted after the log files are moved or deleted so that new log files are opened.

See Also: "Log Rotation" in the Apache Server documentation.

Access Log

Rog records all requests processed by the server. The location and content of the access log is controlled by the CustomLog directive. The `LogFormat` directive can be used to simplify the selection of the contents of the logs.

Specifying LogFormat

`LogFormat` specifies the information included in the log file, and the manner in which it is written. The default format is the Common Log Format (CLF). The CLF format is: `host ident authuser date request status bytes`

- `host`: This is the client domain name or its IP number.
- `ident`: If `IdentityCheck` is enabled and the client machine runs `identd`, then this is the client identity information.
- `authuser`: This is the user ID for authorized user.
- `date`: This is the date and time of the request in the `<day/month/year:hour:minute:second>` format.
- `request`: This is the request line, in double quotes, from the client.
- `status`: This is the three-digit status code returned to the client.
- `bytes`: This is the number of bytes, excluding headers, returned to the client.

See Also: "Access Log" in the Apache Server documentation.

CustomLog

The `CustomLog` directive is used to log requests to the server. A log format is specified, and the logging can optionally be made conditional on request characteristics using environment variables.

See Also: "`CustomLog` directive" in the Apache Server documentation.

Error Log

Sends diagnostic information and records error messages to a log file located, by default, in:

- UNIX: *`ORACLE_HOME`*`/ohs/logs/error_log`
- Windows: *`ORACLE_HOME`*`\ohs\logs\error_log`

The file name can be set using the ErrorLog directive.

See Also: "`ErrorLog` directive" in the Apache Server documentation.

PID File

When the server is started, it notes the process ID of the parent httpd process to the PID file located, by default, in

- UNIX: *`ORACLE_HOME`*`/ohs/logs/httpd.pid`
- Windows: *`ORACLE_HOME`*`\ohs\logs\httpd.pid`

This filename can be changed with the PidFile directive. The process ID is for use by the administrator for restarting and terminating the daemon. If the process dies (or is killed) abnormally, then it is necessary to kill the children httpd processes.

See Also: "Pid File" in the Apache Server documentation.

Piped Log

Oracle HTTP Server is capable of writing error and access log files through a pipe to another process, rather than directly to file. This increases the flexibility of logging, without adding code to the main server. In order to write logs to a pipe, replace the file name with the pipe character "|", followed by the name of the executable which should accept log entries on its standard input. Oracle HTTP Server starts the piped-log process when the server starts, and restarts it if it crashes while the server is running.

Piped log processes are spawned by the parent Oracle HTTP Server httpd process, and inherit the user ID of that process. This means that piped log programs usually run as `root` so it is important to keep the programs simple and secure.

See Also: "Piped Log" in the Apache Server documentation.

Rewrite Log

Rewrite Log is necessary for debugging when mod_rewrite is used. This log file produces a detailed analysis of how the rewriting engine transforms requests. The level of detail is controlled by the `RewriteLogLevel` directive.

See Also: "Rewrite Log" in the Apache Server documentation.

Script Log

Enables you to record the input to and output from the CGI scripts. This should only be used in testing, and not for live servers.

See Also: "Rewrite Log" in the Apache Server documentation.

SSL Log

When Oracle HTTP Server starts in SSL mode, it creates `ssl_engine_log` and `ssl_request_log` in

- UNIX: *`ORACLE_HOME`*`/ohs/logs`
- Windows: *`ORACLE_HOME`*`\ohs\logs`

`ssl_engine_log` tracks SSL and protocol issues, where as `ssl_request_log` records user activity. Use the `SSLLogFile` directive to control output.

See Also: Chapter 9, "Enabling SSL for Oracle HTTP Server"

Transfer Log

Specifies the file in which to store the log of accesses to the site. If it is not explicitly included in the `conf` file, then no log is generated. The server typically logs each request to a transfer file located, by default, in

- UNIX:*`ORACLE_HOME`*`/ohs/logs/access_log`
- Windows: *`ORACLE_HOME`*`\ohs\logs\access_log`

The filename can be set using a CustomLog directive.

7

Understanding Modules

This chapter describes the modules (mods) included in the Oracle HTTP Server. The modules extend the basic functionality of the Web server, and support integration between Oracle HTTP Server and other Oracle Application Server components.

Documentation from the Apache Software Foundation is referenced when applicable.

> **Note:** Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

List of Modules

Table 7–1 lists all the Oracle HTTP Server modules discussed in this chapter.

Table 7–1 Oracle HTTP Server Modules

Oracle HTTP Server Modules			
mod_access	mod_actions	mod_alias	mod_asis
mod_auth	mod_auth_anon	mod_auth_dbm	mod_autoindex
mod_cern_meta	mod_certheaders	mod_cgi	mod_cgid
mod_dir	mod_env	mod_expires	mod_fastcgi
mod_file_cache	mod_headers	mod_imap	mod_include
mod_info	mod_log_config	mod_logio	mod_mime
mod_mime_magic	mod_negotiation	mod_oc4j	mod_onsint
mod_ossl	mod_osso	mod_perl	mod_php
mod_proxy	mod_rewrite	mod_setenvif	mod_speling
mod_status	mod_unique_id	mod_userdir	mod_usertrack
mod_vhost_alias	mod_wchandshake		

mod_access

Controls access to the server based on characteristics of a request, such as hostname or IP address.

See Also: Module `mod_access` in the Apache Server documentation.

mod_actions

Enables execution of CGI scripts based on file type or request method.

See Also: Module `mod_actions` in the Apache Server documentation.

mod_alias

Enables manipulation of URLs in processing requests. It provides mapping between URLs and file system paths, and URL redirection capabilities.

See Also: Module `mod_alias` in the Apache Server documentation.

mod_asis

Enables sending files that contain their own HTTP headers.

See Also: Module `mod_asis` in the Apache Server documentation.

mod_auth

Enables user authentication with files based user lists.

See Also: Module `mod_auth` in the Apache Server documentation.

mod_auth_anon

Enables anonymous user access to protected areas (similar to anonymous FTP, where the email addresses can be logged).

See Also: Module `mod_auth_anon` in the Apache Server documentation.

mod_auth_dbm

Uses DBM files to provide user authentication.

mod_autoindex

Generates directory indexes automatically.

See Also: Module `mod_autoindex` in the Apache Server documentation.

mod_cern_meta

Emulates CERN (Conseil Europeen pour le Recherche Nucleaire) HTTPD metafile semantics. Metafiles are additional HTTP headers that can be produced for each file the server accesses, in addition to the typical set.

mod_certheaders

Allows reverse proxies that terminate SSL connections in front of Oracle HTTP Server, such as OracleAS Web Cache, to transfer information regarding SSL connection, such as SSL client certificate information, to Oracle HTTP Server, and applications running behind Oracle HTTP Server. This information is transferred from the reverse proxy to Oracle HTTP Server using HTTP headers. The information is transferred from the headers to the standard CGI environment variable, which `mod_ossl` or `mod_ssl` populates if the SSL connection is terminated by Oracle HTTP Server. It is an Oracle module.

It also allows certain requests to be treated as HTTPS requests even though they are received through HTTP. This is done using the `SimulateHttps` and `AddCertHeader` directives.

`SimulateHttps` takes the container it is contained within, such as **<VirtualHost>**, **<Location>**, and so on, and treats all requests received for this container as if they were received through HTTPS, regardless of the real protocol that the request was received through.

`AddCertHeader` is specifically for use with OracleAS Web Cache. For OracleAS Web Cache, it adds a special header that indicates to Oracle HTTP Server which requests OracleAS Web Cache received through HTTPS. `mod_certheaders` triggers Oracle HTTP Server to only treat those cases where OracleAS Web Cache received the request as HTTPS as if Oracle HTTP Server received it through HTTPS.

Perform the following steps to configure `mod_certheaders`:

1. Configure Oracle HTTP Server to load `mod_certheaders`. To do this, add a `LoadModule` directive to `httpd.conf` file:
   - UNIX: `LoadModule certheaders_module libexec/mod_certheaders.so`
   - Windows: `LoadModule certheaders_module modules/ApacheModuleCertHeaders.dll`
2. Specify which headers should be translated to CGI environment variables. This can be achieved by using the `AddCertHeader` directive. This directive takes a single argument, which is the CGI environment variable that should be populated from a HTTP header on incoming requests. For example, to populate the `SSL_CLIENT_CERT` CGI environment variable, add the following line to `httpd.conf`:

   ```
   AddCertHeader SSL_CLIENT_CERT
   ```

   The `AddCertHeader` directive can be a global setting if it is placed in the base virtual server section of `httpd.conf`. It can be specific to a single virtual host by placing it within a virtual host container, or it can be specific to a set of URIs by placing it within a **<Directory>** or **<Location>** container directive within `httpd.conf`. The combination of these directives are additive, so that for a given URI, all directives that are specific to that URI will be added to any that are specific to that request's virtual host, which will be added to any that is defined for that base virtual host.

Table 7–2 lists all the supported CGI environment variables with their corresponding HTTP header names.

Table 7–2 CGI Environment Variables with Corresponding Header Names

CGI Variable	Header Name	CGI Variable	Header Name
SSL_PROTOCOL	SSL-Protocol	SSL_SESSION_ID	SSL-Session_Id
SSL_CIPHER	SSL-Cipher	SSL_CIPHER_EXPORT	SSL-Cipher-Export
SSL_CIPHER_ALGKEYSIZE	SSL-Cipher-Algkeysize	SSL_VERSION_LIBRARY	SSL-Version-Library
SSL_CLIENT_CERT	SSL-Client-Cert	SSL_VERSION_INTERFACE	SSL-Version-Interface
SSL_CLIENT_CERT_CHAIN_n	SSL-Client-Cert-Chain-n	SSL_CIPHER_USEKEYSIZE	SSL-Cipher-Usekeysize
SSL_CLIENT_VERIFY	SSL-Client-Verify	SSL_SERVER_CERT	SSL-Server-Cert
SSL_CLIENT_M_VERSION	SSL-Client-M-Version	SSL_SERVER_M_VERSION	SSL-Server-M-Version
SSL_CLIENT_M_SERIAL	SSL-Client-M-Serial	SSL_SERVER_M_SERIAL	SSL-Server-M-Serial
SSL_CLIENT_V_START	SSL-Client-V-Start	SSL_SERVER_V_END	SSL-Server-V-End
SSL_CLIENT_V_END	SSL-Client-V-End	SSL_SERVER_V_END	SSL-Server-V-End
SSL_CLIENT_S_DN	SSL-Client-S-DN	SSL_SERVER_S_DN	SSL-Server-S-DN
SSL_CLIENT_S_DN_C	SSL-Client-S-DN-C	SSL_SERVER_S_DN_C	SSL-Server-S-DN-C
SSL_CLIENT_S_DN_ST	SSL-Client-S-DN-ST	SSL_SERVER_S_DN_ST	SSL-Server-S-DN-ST
SSL_CLIENT_S_DN_L	SSL-Client-S-DN-L	SSL_SERVER_S_DN_L	SSL-Server-S-DN-L
SSL_CLIENT_S_DN_O	SSL-Client-S-DN-O	SSL_SERVER_S_DN_O	SSL-Server-S-DN-O
SSL_CLIENT_S_DN_OU	SSL-Client-S-DN-OU	SSL_SERVER_S_DN_OU	SSL-Server-S-DN-OU
SSL_CLIENT_S_DN_CN	SSL-Client-S-DN-CN	SSL_SERVER_S_DN_CN	SSL-Server-S-DN-CN
SSL_CLIENT_S_DN_T	SSL-Client-S-DN-T	SSL_SERVER_S_DN_T	SSL-Server-S-DN-T
SSL_CLIENT_S_DN_I	SSL-Client-S-DN-I	SSL_SERVER_S_DN_I	SSL-Server-S-DN-I
SSL_CLIENT_S_DN_G	SSL-Client-S-DN-G	SSL_SERVER_S_DN_G	SSL-Server-S-DN-G
SSL_CLIENT_S_DN_S	SSL-Client-S-DN-S	SSL_SERVER_S_DN_S	SSL-Server-S-DN-S
SSL_CLIENT_S_DN_D	SSL-Client-S-DN-D	SSL_SERVER_S_DN_D	SSL-Server-S-DN-D
SSL_CLIENT_S_DN_UID	SSL-Client-S-DN-Uid	SSL_SERVER_S_DN_UID	SSL-Server-S-DN-Uid
SSL_CLIENT_S_DN_Email	SSL-Client-S-DN-Email	SSL_SERVER_S_DN_Email	SSL-Server-S-DN-Email
SSL_CLIENT_I_DN	SSL-Client-I-DN	SSL_SERVER_I_DN	SSL-Server-I-DN
SSL_CLIENT_I_DN_C	SSL-Client-I-DN-C	SSL_SERVER_I_DN_C	SSL-Server-I-DN-C
SSL_CLIENT_I_DN_ST	SSL-Client-I-DN-ST	SSL_SERVER_I_DN_ST	SSL-Server-I-DN-ST
SSL_CLIENT_I_DN_L	SSL-Client-I-DN-L	SSL_SERVER_I_DN_L	SSL-Server-I-DN-L
SSL_CLIENT_I_DN_O	SSL-Client-I-DN-O	SSL_SERVER_I_DN_O	SSL-Server-I-DN-O
SSL_CLIENT_I_DN_OU	SSL-Client-I-DN-OU	SSL_SERVER_I_DN_OU	SSL-Server-I-DN-OU
SSL_CLIENT_I_DN_CN	SSL-Client-I-DN-CN	SSL_SERVER_I_DN_CN	SSL-Server-I-DN-CN
SSL_CLIENT_I_DN_T	SSL-Client-I-DN-T	SSL_SERVER_I_DN_T	SSL-Server-I-DN-T

Table 7–2 (Cont.) CGI Environment Variables with Corresponding Header Names

CGI Variable	Header Name	CGI Variable	Header Name
SSL_CLIENT_I_DN_I	SSL-Client-I-DN-I	SSL_SERVER_I_DN_I	SSL-Server-I-DN-I
SSL_CLIENT_I_DN_G	SSL-Client-I-DN-G	SSL_SERVER_I_DN_G	SSL-Server-I-DN-G
SSL_CLIENT_I_DN_S	SSL-Client-I-DN-S	SSL_SERVER_I_DN_S	SSL-Server-I-DN-S
SSL_CLIENT_I_DN_D	SSL-Client-I-DN-D	SSL_SERVER_I_DN_D	SSL-Server-I-DN-D
SSL_CLIENT_I_DN_UID	SSL-Client-I-DN-Uid	SSL_SERVER_I_DN_UID	SSL-Server-I-DN-Uid
SSL_CLIENT_I_DN_Email	SSL-Client-I-DN-Email	SSL_SERVER_I_DN_Email	SSL-Server-I-DN-Email
SSL_CLIENT_A_SIG	SSL-Client-A-Sig	SSL_SERVER_A_SIG	SSL-Server-A-Sig
SSL_CLIENT_A_KEY	SSL-Client-A-Key	SSL_SERVER_A_KEY	SSL-Server-A-Key

3. `mod_certheaders` can be used to instruct Oracle HTTP Server to treat certain requests as if they were received through HTTPS even though they were received through HTTP. This is useful when Oracle HTTP Server is front-ended by a reverse proxy or load balancer, which acts as a termination point for SSL requests, and forwards the requests to Oracle HTTP Server through HTTPS.

   If OracleAS Web Cache is being used as the load balancer, it sends an HTTP header that identifies all requests it received through HTTPS. This means that `mod_certheaders` automatically detects which requests should be treated as HTTPS requests by simply looking for this header. To enable this, add the following directive to `httpd.conf`:

   ```
   AddCertHeader HTTPS
   ```

   This affects all URLs processed by Oracle HTTP Server.

   For other load balancers, `mod_certheaders` must be explicitly configured to determine which requests should be treated as HTTPS requests. To do this, use the following directive:

   ```
   SimulateHttps on
   ```

   `SimulateHttps` can be embedded within a virtual host, such as:

   ```
   <VirtualHost localhost:7777>
       SimulateHttps on
       .
       .
       .
   </VirtualHost>
   ```

   This tells `mod_certheaders` to treat every request handled by this virtual host as HTTPS, or the directive can be placed within a **<LocationMatch>**, **<Directory>**, or **<DirectoryMatch>** directive container such as:

   ```
   <Location /foo/>
       SimulateHttps on
   </Location>
   ```

   This limits it to URLs starting with `/foo/`.

4. Edit the $ORACLE_HOME/sso/conf/sso_apache.conf, and comment out the following line:

```
#SSLOptions +ExportCertData +StdEnvVars
```

5. Runthe following command:

   ```
   dcmctl updateconfig -ct ohs
   ```

6. Run the following command:

   ```
   opmnctl restartproc type=ohs
   ```

7. Test that the SSO server can be logged into with client authentication.

mod_cgi

Enables the server to run CGI scripts.

> **See Also:** Module `mod_cgi` in the Apache Server documentation.

mod_cgid

Except for the optimizations and the additional `ScriptSock` directive, this module functions similarly to mod_cgi. The `ScriptSock` directive sets the name of the socket to use for communication with the CGI daemon. The socket will be opened using the permissions of the user who starts Apache (usually root).

> **See Also:** Module `mod_cgid` in the Apache Server documentation.

mod_dir

Enables the server to perform slash (/) redirects. Directories must contain a trailing slash. If a request for a URL without a trailing slash is received, `mod_dir` redirects the request to the same URL followed by a trailing slash. For example:

```
http://myserver/documents/mydirectory
```

is redirected to

```
http://myserver/documents/mydirectory/
```

> **See Also:** Module `mod_dir` in the Apache Server documentation.

mod_env

Enables you to control the environment for CGI scripts and SSI (Server Side Includes) pages by passing, setting, and unsetting environment variables.

`ModifyEnv` appends or prepends a value to an existing `ENV` variable's value, and passes it into the Oracle HTTP Server environment. The following is the usage:

Let `$FOO` = `"foo"`:

```
ModifyEnv FOO "bar" modifies the value of $FOO from "foo" to "foo:bar"

ModifyEnv FOO "+bar" modifies the value of $FOO from "foo" to "bar:foo"
```

Let `$FOO` be undefined:

```
Modify Foo "bar" sets the value of $FOO to "bar"
```

See Also: Module `mod_env` in the Apache Server documentation.

mod_expires

Enables the server to generate Expires HTTP headers, which provide information to the client about document validity. Documents are served from the source if, based on the expiration criteria, the cached copy has expired.

See Also: Module `mod_expires` in the Apache Server documentation.

mod_fastcgi

Supports the FastCGI protocol, which enables you to maintain a pool of running servers for CGI applications, thereby eliminating start-up and initialization overhead.

See Also: Module `mod_fastcgi` in the Apache Server documentation.

mod_file_cache

Provides techniques for caching frequently requested static files. This module should be used with care as you can easily create a broken site.

See Also: Module `mod_file_cache` in the Apache Server documentation.

mod_headers

Enables you to merge, replace, or remove HTTP response headers.

See Also: Module `mod_headers` in the Apache Server documentation.

mod_imap

Enables server-side image map processing.

mod_include

Provides a filter that processes documents for SSI (Server Side Includes) directives.

See Also: Module `mod_include` in the Apache Server documentation.

mod_info

Summarizes the entire server configuration, including all installed modules and directive settings.

See Also: Module `mod_info` in the Apache Server documentation.

mod_log_config

Provides configurable, customizable logging of server activities. You can choose the log format, and select or exclude individual requests for logging, based on characteristics of the requests.

> **See Also:** Module `mod_log_config` in the Apache Server documentation.

mod_logio

Provides the logging of input and output numbers of bytes received and sent per request. The numbers reflect the actual bytes as received on the network, which then takes into account the headers and bodies of requests and responses.

> **See Also:** Module `mod_logio` in the Apache Server documentation.

mod_mime

Enables the server to determine the type of a file from its filename, and associate files with handlers for processing.

> **See Also:** Module `mod_mime` in the Apache Server documentation.

mod_mime_magic

Enables the server to determine the MIME type of a file by examining a few bytes of its content. It is used in cases when mod_mime cannot determine a file type. Make sure that `mod_mime` appears before `mod_mime_magic` in the configuration file, so that `mod_mime` processes the files first.

> **See Also:** Module `mod_mime_magic` in the Apache Server documentation.

mod_negotiation

Enables the server for content negotiation (selection of documents based on the client's capabilities).

> **See Also:** Module `mod_negotiation` in the Apache Server documentation.

mod_oc4j

Routes requests from the Oracle HTTP Server to Oracle Application Server Containers for J2EE (OC4J), through the AJP 1.3 protocol. It is an Oracle module.

`mod_oc4j` is enabled by default. During installation, the `oc4j_deploy_tool.jar` adds mount points to mod_oc4j.conf for applications deployed into OC4J instances. Requests that come in for specific mount points in `mod_oc4j` are routed to the OC4J instance for that mount point.

OC4J instances are started and managed by Oracle Process Manager and Notification Server (OPMN).

See Also:

- *Oracle Application Server Containers for J2EE User's Guide*
- *Oracle Process Manager and Notification Server Administrator's Guide*

This section discusses the following topics:

- Configuring mod_oc4j
- Load Balancing Using mod_oc4j
- Enabling SSL between mod_oc4j and OC4J

Configuring mod_oc4j

The following sections describe all relevant directives in httpd.conf and mod_oc4j.conf. Sample configurations are also provided.

mod_oc4j Configuration File and Directives

The `mod_oc4j` directives are maintained in `mod_oc4j.conf`. The `mod_oc4j.conf` file is included by default into the `httpd.conf` file, using the following directive:

```
include "ORACLE_HOME/ohs/conf/mod_oc4j.conf"
```

The following directives are used to configure `mod_oc4j`:

- Oc4jCacheSize
- Oc4jConnTimeout
- Oc4jCookieExtension
- Oc4jExtractSSL
- Oc4jEnvVar
- Oc4jMount
- Oc4jMountCopy
- Oc4jUseOHSErrors

See Also: "Using SSL Configuration Directives" on page 9-4

LoadModule

Loads the `mod_oc4j` module.

Category	Value
Syntax	`LoadModule oc4j_module` *`mod_oc4j shared library file`*
Required	Yes
Default	■ UNIX: None ■ Windows: `LoadModule oc4j_module modules\ApacheModuleOc4j.dll`
Example	■ UNIX: `LoadModule oc4j_module mod_oc4j.so` ■ Windows: `LoadModule oc4j_module modules\ApacheModuleOc4j.dll`

Oc4jCacheSize

Specifies the size of the OC4J connection cache.

Category	Value
Syntax	`Oc4jCacheSize` *<size of connection cache>*
Required	No
Default	■ UNIX: 1 ■ Windows: 32
Example	`Oc4jCacheSize 64`
Usage	Specifies the number of concurrent OC4J connections that can be cached by each Oracle HTTP Server process. Setting this directive to "0" will disable persistent connections between `mod_oc4j` and the OC4J instances.

Oc4jConnTimeout

Defines maximum idle time (in seconds) for unused connections.

Category	Value
Syntax	`Oc4jConnTimeout` *<timeout value for AJP13 connections>*
Required	No
Default	None
Example	`Oc4jConnTimeout` 10
Usage	Useful for cases where there is a firewall between `mod_oc4j` and OC4J that times out connections. The value should be set to a value smaller than the timeout value used by the firewall.

Oc4jCookieExtension

Directs `mod_oc4j` to use `JSESSIONID_`*<cookie_name_extension>* as OC4J's session identifier in the cookie.

Category	Value
Syntax	`Oc4jCookieExtension` *<cookie_name_extension>*
Required	No
Default	None
Example	`Oc4jCookieExtension MYEXT`
Usage	Directs `mod_oc4j` to use `JSESSIONID_`*<cookie_name_extension>* as OC4J's session identifier in the cookie, instead of `JSESSIONID`. In the preceding example, `JSESSIONID_MYEXT` is used as the OC4J's session identifier.

Oc4jExtractSSL

Governs passing SSL environment variables.

Category	Value
Syntax	`Oc4jExtractSSL` *On|Off*
Required	No

Category	Value
Default	`Off`
Example	`Oc4jExtractSSL On`
Usage	Directs `mod_oc4j` to decide whether or not to pass three SSL environment variables, `SSL_CLIENT_CERT`, `SSL_CIPHER`, and `SSL_SESSION_ID` to OC4J. There is a performance cost associated with copying the SSL environment variables to OC4J, so set it to "On" only if the environment variables must be available to OC4J.

Note: If configured, `mod_oc4j` passes some security environment parameters to OC4J set by `mod_ossl` and `mod_osso`, at request time.

Oc4jEnvVar

Directs `mod_oc4j` to pass some environment variables from Oracle HTTP Server to OC4J.

Category	Value
Syntax	`Oc4jEnvVar` *`environment variable name [environment variable default value]`*
Required	No
Default	None

Example	`Oc4jEnvVar MY_ENV1`
	`Oc4jEnvVar MY_ENV2 myenv_value`

Usage	For each `OC4jEnvVar` entry, you must also configure the Oracle HTTP Server directive, `PassEnv`, with the environment variable. Otherwise, `mod_oc4j` cannot acquire and pass the value. Multiple entries are allowed. You could specify the default value for the environment variable as the second parameter, or leave it empty. If the environment variable's value is found in the Oracle HTTP Server environment, its value will be passed to OC4J. Otherwise, if the default value is set, the default value will be passed. If this environment variable's value is not found in the Oracle HTTP Server environment and the default value is not set, nothing is passed to OC4J. There is a performance degradation associated with `mod_oc4j` passing some configured environment variables over to OC4J with each request.

Note: If configured, `mod_oc4j` passes some security environment parameters to OC4J set by `mod_ossl` and `mod_osso`, at request time.

Oc4jMount

Directs `mod_oc4j` to route requests containing a particular path to a destination. A destination can be a single OC4J process, or a set of OC4J instances.

Category	Value

Syntax	`Oc4jMount` *`path [destination]`*
	where path is the context root. The path parameter must be the same as the application context root specified in the OC4J configuration file `xxx-web-site.xml`. The application context root is shown in bold text in the example <web-site> element.
	<default-web-app application="default" name="defaultWebApp" root="/j2ee"/>
	and where destination is one of these types:
	- `ajp13_dest`
	- `cluster_dest` (this is the default destination type)
	- `instance_dest`
	If destination is not specified, the default OC4J instance name of `home` will be used. For example,
	`Oc4jMount /myApp/*`
	provides the same result as:
	`Oc4jMount /myApp/* cluster://local_ias_cluster_name:home`

Required	No
Default	None

Examples	`Oc4jMount /app01/* ajp13://my-sun:8888`
	`Oc4jMount /app02/*`
	`Oc4jMount /app03/* home`
	`Oc4jMount /app04/* ias_cluster_1:home`
	`Oc4jMount /app05/* cluster://ias_cluster_1:home,ias_cluster_2:home`
	`Oc4jMount /app06/* instance://ias_instance_1:home`
	`Oc4jMount /app07/* instance://ias_instance_1:home_1,ias_instance_2:home_2`
	`Oc4jMount /app08/* instance://my-sun:ias_instance_1:home`

Category	Value
Usage	Examples are provided for each routing destination: **`ajp13_dest`** `Oc4jMount` path ajp13://*my-sun:8888* A request with the pattern specified in path is routed to an OC4J process listening on *my-sun*, port 8888 with the AJP 1.3 protocol. (*my-sun* and port 8888 are the AJP 1.3 protocol host and port specified in the OC4J configuration file `xxx-web-site.xml`. **`cluster_dest`** `Oc4jMount` <*path*> cluster: *`//ias_cluster_name:OC4J_instance_name, ias_ cluster_name:OC4J_instance_name...`* A request with the pattern specified in path is load balanced to one or more of the OC4J instances specified (instances are separated by commas). The Oracle Application Server Cluster Name is optional. If it is provided, the destination OC4J instance should be inside the named cluster. If none is provided, the destination OC4J instance should be inside the local Oracle Application Server cluster. **`instance_dest`** `Oc4jMount` <*path*> instance: *`//host:ias_instance_name:OC4J_instance_name, host:ias_instance_name:OC4J_instance_name...`* A request with the pattern specified in <*path*> is load balanced to one or more of the OC4J instances specified (instances are separated by commas). The host name is optional. If it is provided, the destination OC4J instance should be inside the Oracle Application Server instance residing on that host. If none is provided, the destination OC4J instance could be on any host.

Oc4jMountCopy

Copies mount points from the base server.

Category	Value
Syntax	`Oc4jMountCopy` *`On|Off`*
Required	No
Default	On
Example	`Oc4jMountCopy Off`
Usage	Directs `mod_oc4j` to decide whether to copy `Oc4jMount` points from the base server to the virtual host on which this directive is specified. If its value is On, all of the `Oc4jMount` points configured in the base server will be copied to the virtual host. If its value is `Off`, only the `Oc4jMount` points configured within the virtual host scope will be used.

Oc4jUseOHSErrors

Allows users to configure an error range using Oracle HTTP Server's error pages when errors in the range are returned from OC4J.

Category	Value
Syntax	`Oc4jUseOHSErrors` *`On|Off/min-max`*
Required	No

Category	Value
Default	off
Example	`OC4jUseOHSErrors 400-410`
Usage	`Oc4jUseOHSErrors Off`: This is the default value if `Oc4jUseOHSErrors` is not specified. OC4J error pages are passed back to the client for all error values. `Oc4jUseOHSErrors on`: This returns the Oracle HTTP Server error pages for HTTP errors 400-500 inclusive. `Oc4jUseOHSErrors min-max`: This specifies the min and max for HTTP errors. For example, if you set `Oc4jUseOHSErrors 400-410`, then Oracle HTTP Server error pages for HTTP error 400-410 inclusive are returned from OC4J.

mod_oc4j Sample Configurations

This section provides some sample configurations for `mod_oc4j`.

Example 7–1 Sample mod_oc4j configuration

This configuration mounts all requests starting with the URI /servlet/ to the default instance of OC4J processes.

Make this entry in the `httpd.conf` file:

```
Oc4jMount /servlet/*
```

Example 7–2 Sample mod_oc4j configuration

This configuration performs the same work as the configuration in Example 7–1, using a `<Location>` container directive instead of the `Oc4jMount` directive.

Make this entry in the `httpd.conf` file:

```
<Location /servlet>
    SetHandler oc4j-handler
</Location>
```

Note: This will only route requests to default the OC4J instance

Example 7–3 Sample mod_oc4j configuration

This configuration mounts all requests starting with the URI `/servlet/` or `/j2ee/` and all JSP pages to the default OC4J instance of OC4J processes.

Make these entries in the `mod_oc4j.conf` file:

```
Oc4JMount /servlet/*
Oc4JMount /*.jsp
Oc4JMount /j2ee/*
```

Example 7–4 Sample mod_oc4j configuration

This configuration mounts:

- All requests starting with the URI `/applicationA/` and all JSP pages to `oc4j_instance_A`, in which all OC4J processes are managed by OPMN.

- All requests starting with the URI /applicationB/ to oc4j_instance_B, in which all OC4J processes are managed by OPMN.

  Make these entries in the mod_oc4j.conf file:

  ```
  Oc4JMount /applicationA/* oc4j_instance_A
  Oc4JMount /applicationB/* oc4j_instance_B
  Oc4JMount /j2ee/*
  Oc4JMount /*.jsp oc4j_instance_A
  ```

Load Balancing Using mod_oc4j

mod_oc4j load balancing, including metric based load balancing, is discussed in detail in Appendix A, "Load Balancing Using mod_oc4j".

Enabling SSL between mod_oc4j and OC4J

Optionally, you can have direct SSL support for communication between mod_oc4j and OC4J. To do this, you have to enable SSL on the mod_oc4j side as well as the OC4J side.

- Enabling SSL for mod_oc4j
- Enabling SSL for OC4J

Enabling SSL for mod_oc4j

Add the following directives in mod_oc4j.conf to enable SSL for mod_oc4j:

Oc4jEnableSSL

Indicates whether mod_oc4j needs to use SSL when communicating with OC4J processes. It should not be configured to "On" if Oc4jiASPTActive is configured to "On".

Category	Value
Parameter Name	Oc4jEnableSSL
Parameter Type	string
Valid Values	On/Off
Default Value	Off

Oc4jSSLWalletFile

When Oc4jEnableSSL is set to "On", this directive specifies the location of an Oracle Wallet file that contains SSL certificates that are used for SSL communication with OC4J processes.

Category	Value
Parameter Name	Oc4jSSLWalletFile
Parameter Type	string
Valid Values	Path to a wallet directory location that contains the SSL certificate to be used when establishing SSL connections to OC4J processes.
Default Value	N/A

Oc4jSSLWalletPassword

When Oc4jEnableSSL is set to "On", this value is the obfuscated password used for authentication when opening the wallet file. This value is obtained using the utility provided with the Oracle Wallet Manager.

Category	Value
Parameter Name	`Oc4jSSLWalletPassword`
Parameter Type	string
Valid Values	Obfuscated password used for authentication when opening the wallet file specified by Oc4jSSLWalletFile.
Default Value	N/A

See Also:

- *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.
- "Using the iasobf Utility" on page 9-15

Note: Wallet passwords have been deprecated. A warning message is generated in the Oracle HTTP Server log if this directive is used. For secure wallets, Oracle recommends that you get a SSO wallet instead.

Enabling SSL for OC4J

To enable SSL communication between `mod_oc4j` and OC4J, you have to enable SSL on the OC4J side too.

See Also: *Oracle Application Server Containers for J2EE Security Guide* for enabling SSL on the OC4J side.

mod_onsint

Provides integration support with Oracle Notification Service (ONS) and Oracle Process Manager and Notification Server (OPMN). It is an Oracle module.

Benefits of mod_onsint

`mod_onsint` provides the following functionality:

- Provides a subscription mechanism for ONS notifications within Oracle HTTP Server. This is particularly important on UNIX where Oracle HTTP Server employs a multi-process architecture. In such an architecture, it is not feasible to have an ONS subscriber in each process since there are up to 8192 processes that comprise a single Oracle HTTP Server instance. Instead, `mod_onsint` provides a single process that receives notification for all modules within an Oracle HTTP Server instance.
- Publishes `PROC_READY` ONS notifications so that other components such as OPMN and OC4J are notified that the listener is up and ready. It also provides information such as DMS metrics and information about how the listener can be contacted. These notifications are sent periodically by `mod_onsint` as long as the Oracle HTTP Server instance is running.

- Provides functionality that allows Oracle HTTP Server to terminate as a single unit if the parent process fails. The parent process is responsible for starting and stopping all of the child processes for an Oracle HTTP Server instance. The failure of the parent process without first shutting down the child processes leaves Oracle HTTP Server in an inconsistent state that can only be fixed by manually killing all of the orphaned child processes. Until this is done, a new Oracle HTTP Server instance cannot be started since the orphaned child processes still occupy the ports Oracle HTTP Server wants to use. `mod_onsint` provides a monitor of the parent process. If it detects that the parent process has died, it kills all of the remaining child processes. When combined with OPMN, this provides restartability for Oracle HTTP Server in the case of a parent process failure. `mod_onsint` ensures that all of the Oracle HTTP Server child processes die, leaving the ports open for a new Oracle HTTP Server instance. OPMN ensures that a new instance is started once the failure of the original instance is detected.

Implementation Differences on UNIX and Windows

Due to the difference in architecture of Oracle HTTP Server on UNIX and Windows, the implementation of `mod_onsint` varies slightly on these platforms.

On UNIX, `mod_onsint` spawns a process at module initialization time. This process is responsible for watching the parent process as well as sending and receiving ONS messages. Callback functions from other modules interested in ONS notifications are made in this process. For this information to be shared with other Oracle HTTP Server child processes, the use of an interprocess communication method such as a memory mapped file must be used. If a failure of a parent process is detected on UNIX, a signal is sent to all the other child processes, causing them to shut down.

On Windows, Oracle HTTP Server consists of only two processes, the parent and a multi-threaded child that handles all of the HTTP requests. In this model, `mod_onsint` runs as a thread within the child process. This thread watches the parent process as well as sending and receiving ONS messages. Callback functions from other modules interested in ONS notifications are made in the child process. If a failure of the parent process is detected, the `mod_onsint` terminates the child process, effectively shutting down Oracle HTTP Server.

There is an optional directive called `OpmnHostPort` that can be configured for `mod_onsint`. This directive enables you to specify a hostname and port that OPMN should use for pinging the Oracle HTTP Server instance that `mod_onsint` is running in. If `OpmnHostPort` is not specified, `mod_onsint` chooses an HTTP port automatically. In certain circumstances, you may want to choose a specific HTTP port and hostname that OPMN should use to ping the listener with.

`OpmnHostPort` takes a single argument which is a `host:port` string that specifies the values to pass to OPMN. For example, the following line would specify that OPMN should use the localhost interface and port 7778 to ping this listener:

```
OpmnHostPort localhost: 7778
```

This directive must be in the global section of the httpd.conf file. It cannot be embedded into any virtual host of location container. After installation, an `OpmnHostPort` directive is located in dms.conf. It points OPMN to the Oracle HTTP Server "diagnostic port", which is a special localhost only virtual host.

mod_ossl

Enables strong cryptography for Oracle HTTP Server. This Oracle module is a plug-in to Oracle HTTP Server that enables the server to use SSL. It is very similar to the

OpenSSL module, `mod_ssl`. However, in contrast to the OpenSSL module, `mod_ossl` is based on the Oracle implementation of SSL, which supports SSL, version 3, and is based on Certicom and RSA Security technology.

> **See Also:**
>
> - *Oracle Application Server Security Guide*
> - "Using mod_ossl to Authenticate Users" on page 8-7
> - Chapter 9, "Enabling SSL for Oracle HTTP Server"

mod_osso

Enables **single sign-on** for Oracle HTTP Server. `mod_osso` examines incoming requests and determines whether the resource requested is protected, and if so, retrieves the Oracle HTTP Server cookie for you. It is an Oracle module.

> **See Also:** *Oracle Application Server Single Sign-On Administrator's Guide*

mod_perl

Embeds the Perl interpreter into the Oracle HTTP Server. This eliminates start-up overhead and enables you to write modules in Perl. Oracle Application Server uses Perl version 5.8.3.

> **See Also:** `mod_perl` Guide

Database Usage Notes

This section provides information for `mod_perl` users working with databases. It explains how to test a local database connection and set character forms.

Using Perl to Access the Database

The following section contains information about using Perl to access the database. Perl scripts access databases using the DBI/DBD driver for Oracle. The DBI/DBD driver is part of Oracle Application Server. It calls Oracle Callable Interface (OCI) to access the databases.

DBI must be enabled in httpd.conf for DBI to function. To do this, perform the following steps:

1. Edit `httpd.conf` using a text editor.
2. Search for "`PerlModule Apache::DBI`".
3. Uncomment the line `PerlModule Apache::DBI`".
4. Restart Oracle HTTP Server using the following command:
    - UNIX: `ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`
    - Windows: `ORACLE_HOME\opmn\bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`

Files must be copied to `ORACLE_HOME/ohs/cgi-bin`

Example 7–5 Using Perl to Access the Database

```
#!<ORACLE_HOME>/perl/bin/perl -w
  use DBI;
  my $dataSource = "host=<hostname.domain>;sid=<orclsid>;port=1521";
  my $userName = "scott";
  my $password = "tiger";
  my $dbhandle = DBI->connect("dbi:Oracle:$dataSource", $userName, $password)
    or die "Can't connect to the Oracle Database: $DBI::errstr\n";
  print "Content-type: text/plain\n\n";
  print "Database connection successful.\n";
  ### Now disconnect from the database
  $dbhandle->disconnect
    or warn "Database disconnect failed; $DBI::errstr\n";
  exit;
```

You can access the DBI scripts from the following locations:

```
http://<hostname.domain>:<port>/cgi-bin/<scriptname>
http://<hostname.domain>:<port>/perl/<scriptname>
```

If the script specifies "`use Apache::DBI`" instead of "`use DBI`", then it will only be able to run from `http://`<hostname.domain>`:`<port>`/perl/`<scriptname>.

Testing Database Connection

The following is a sample Perl script for testing the database connection of a local seed database. To use the script to test another database connection, you must replace `scott/tiger` with the user name and password for the target database.

Example 7–6 Sample Perl Script For Testing Connection for Local Seed Database

```
##### Perl script start ######
use DBI;
print "Content-type: text/plain\n\n";
$dbh = DBI->connect("dbi:Oracle:", "scott/tiger", "") || die $DBI::errstr;
 $stmt = $dbh->prepare("select * from emp order by empno")|| die $DBI::errstr;
$rc = $stmt->execute() || die $DBI::errstr;
while (($empno, $name) = $stmt->fetchrow()) { print "$empno $name\n"; }
warn $DBI::errstr if $DBI::err;
die "fetch error: " . $DBI::errstr if $DBI::err;
$stmt->finish() || die "can't close cursor";
$dbh->disconnect() || die "cant't log off Oracle";
##### Perl script End ######
```

Using SQL NCHAR Datatypes

`SQL NCHAR` datatypes have been refined since Oracle9*i*, and are now called reliable Unicode datatypes. `SQL NCHAR` datatypes such as `NCHAR`, `NVARCHAR2` and `NCLOB` allow you to store any Unicode characters regardless of the database character set. The character set for those datatypes is specified by the national character set, which is either AL16UTF-16 or UTF8.

> **See Also:** Oracle9*i* documentation for more about `SQL NCHAR` datatypes.

This release of `DBD::Oracle` supports `SQL NCHAR` datatypes and provides driver extension functions to specify the character form for data binding. The following script shows an example to access `SQL NCHAR` data:

Example 7–7 Sample Script to Access SQLNCHAR Data

```
# declare to use the constants for character forms
use DBD::Oracle qw(:ora_forms);
# connect to the database and get the database handle
$dbh = DBI->connect( ... );
# prepare the statement and get the statement handle
$sth = $dbh->prepare( 'SELECT * FROM TABLE_N WHERE NCOL1 = :nchar1' );
# bind the parameter of a NCHAR type
$sth->bind_param( ':nchar1', $param_1 );
# set the character form to NCHAR
$sth->func( { ':nchar1' => ORA_NCHAR } , 'set_form' );
$sth->execute;
```

As shown in Example 7–7, the set_form function is provided as a private function that you can invoke with the standard DBI `func()` method. It takes an anonymous hash that specifies which placeholder should be associated with which character form. The valid values of character form are either `ORA_IMPLICIT` or `ORA_NCHAR`. Setting the character form to `ORA_IMPLICIT` causes the application's bound data to be converted to the database character set, and `ORA_NCHAR` to the national character set. The default form is `ORA_IMPLICIT`.

Another function is provided to specify the default character set form as follows:

```
# specify the default form to be NCHAR
$dbh->func( ORA_NCHAR, 'set_default_form' );
```

After this call is made, the form of all parameters is `ORA_NCHAR`, unless otherwise specified with `set_form` calls. Note that unlike the `set_form` function, this is a function on the database handle, so every statement from the database handle with its default form specified has the form of your choice by default.

set_form This function sets the character form for parameter(s). Valid forms are either `ORA_IMPLICIT` (default) or `ORA_NCHAR`. The constants are available as: `ora_forms` in `DBD::Oracle`.

Example 7–8 Sample for set_form

```
# a declaration example for the constants ORA_IMPLICIT and ORA_NCHAR
use DBD::Oracle qw(:ora_forms);
# set the character form for the placeholder :nchar1 to NCHAR
$sth->func( { ':nchar1' => ORA_NCHAR } , 'set_form' );
# set the character form using the positional index
$sth->func( { 2 => ORA_NCHAR } , 'set_form' );
# set the character form for multiple placeholders at once
$sth->func( { 1 => ORA_NCHAR, 2 => ORA_NCHAR } , 'set_form' );
```

set_default_form This function sets the default character form for a database handle.

Example 7–9 Default Character Form for a Database Handle

```
$dbh->func( ORA_NCHAR , 'set_default_form' );
```

mod_php

PHP (recursive acronym for "PHP: Hypertext Preprocessor") is an open source, widely-used, general-purpose, client-side scripting language, that is embedded in standard HTML. It is used to generate dynamic HTML pages. On Oracle HTTP Server, PHP support is provided through mod_php and has Oracle database support enabled. It uses PHP version 4.3.9.

Note: phpinfo() prints out very sensitive information about the current state of PHP and Oracle HTTP Server intervals. Users new to PHP, or those who are unaware of phpinfo() should not inadvertantly leave a PHP script called phpinfo() publically accessible.

phpinfo() is used heavily for debugging. There is a good chance that such a debug script could be left in the open by mistake once debugging is finished.

See Also:

- http://php.net/
- "Using PHP with Oracle HTTP Server (OHS)" document on http://www.oracle.com/technology/tech/opensource/index.html if you want to build from the source, or need more information.

mod_proxy

Orovides proxy capability for FTP, CONNECT (for SSL), HTTP/0.9, HTTP/1.0, and HTTP/1.1.

See Also:

- Module mod_proxy in the Apache Server documentation.
- "Using mod_proxy Directives" on page 9-16

mod_rewrite

Oracle HTTP Server provides mod_rewrite as a tool for URL manipulation. A rewriting engine based on a regular-expression parser is used by mod_rewrite to rewrite requested URLs. The granularity of URL manipulations can be affected by the formats of server variables, environment variables, HTTP headers, and time stamps.

This module operates on the full URLs (including the path-info part) both in per-server context (httpd.conf) and per-directory context (.htaccess) and can generate query-string parts on result.

The following topics are discussed in subsequent sections:

- mod_rewrite Rules Processing
- mod_rewrite Directives
- Rewrite Rules Hints
- Redirection Examples

mod_rewrite Rules Processing

Apache processes HTTP in phases. A hook for each of these phases is provided by the Apache API. mod_rewrite uses two of these hooks- the URL-to-filename translation hook which is used after the HTTP request has been read but before any authorization starts, and the Fixup hook which is triggered after the authorization phases and after the per-directory configuration files (.htaccess) have been read, but before the content handler is activated.

mod_rewrite reads the configured rulesets from its configuration structure. Server level rulesets are best configured at startup, while directory level rulesets are configured during the directory access of the kernel.

mod_rewrite loops through the ruleset rule by rule (RewriteRule directive) and when a particular rule matches, it loops through corresponding conditions (RewriteCond directives). First the URL is matched against the Pattern of each rule. When it fails, mod_rewrite looks for corresponding rule conditions. If none are present, it just substitutes the URL with a new value which is constructed from the string Substitution and goes on with its rule-looping. But if conditions exist, it starts an inner loop for processing them in the order that they are listed.

For conditions, a string TestString is created by expanding variables, back-references map lookups, and then CondPattern is matched against the expanded TestString. If the pattern does not match, the complete set of conditions and the corresponding rule fails. If the pattern matches, then the next condition is processed until no more conditions are available. If all conditions match, processing is continued with substituting the URL using Substitution.

When request seeks a URL with more than one slash (/), for example, http://*yourserver*//oldpath/rqstdrsrc, the "//oldpath" may bypass RewriteCond and RewriteRule directives if they are not correctly written.

For example, consider the following rule:

```
RewriteRule ^/oldpath(.*) /newpath$1 [R]
```

Requesting http://*yourserver*/oldpath/files will redirect and return the page http://*yourserver*/newpath/files as expected.

However, requesting http://*yourserver*//oldpath/files will bypass this particular rule, potentially serving a page that you were not expecting it to. You can work around the problem by making sure that rules will capture more than one slash (/). To fix the precedingexample, you should use this replacement:

```
RewriteRule ^/+somepath(.*) /otherpath$1 [R]
```

mod_rewrite Directives

This section discusses the following mod_rewrite directives:

- RewriteEngine
- RewriteOptions
- RewriteLog
- RewriteLogLevel
- RewriteBase

RewriteEngine

Enables or disables the runtime rewriting engine. If it is set to "Off", this module does no runtime processing at all. Use this directive to disable the module instead of commenting out all the RewriteRule directives.

Rewrite configurations are not inherited by default. This means that you need to have `ReWriteEngine On` directive for each virtual host in which you want to use it.

RewriteOptions

By specifying `RewriteOptions` 'inherit', you can force the configuration of the parent by the children. In virtual-server context this means that the maps, conditions and rules of the main server are inherited. In directory context this means that conditions and rules of the `.htaccess` configuration of the parent directory are inherited.

RewriteLog

Sets the name of the file to which the server logs any rewriting action that it performs. If the name does not begin with a slash (/), then it is assumed to be relative to the `Server Root`. To disable logging, either remove or comment out the `RewriteLog` directive or use `RewriteLogLevel 0`. Avoid setting the filename to `/dev/null` to prevent logging. This can slow down the server with no advantage.

RewriteLogLevel

Sets the verbosity level of the rewriting log file. The default level 0 means no logging, while 9 or more means that practically all actions are logged.

RewriteBase

Explicitly sets the base URL for pre-directory rewrites. Rewrite rule can be used in per-directory configuration (`.htaccess`) files. When a substitution occurs for a new URL, the base URL should be added into the server processing. To be able to do this, the module needs to know what the corresponding URL-prefix or URL-base is. By default, this prefix is the corresponding file path itself. However, at most Web sites, URLs are not directly related to physical filename paths. In such cases, you have to use the `RewriteBase` directives to specify the correct URL-prefix.

If the URLs of your Web server are not directly related to physical file paths, you have to use `RewriteBase` in every `.htaccess` files where you want to use `RewriteRule` directives.

Example 7–10 RewriteBase Directive

Assume the following per-directory configuration file:

```
## /abc/def/.htaccess - - per-dir config file for directory /abc/def
 # /abc/def is the physical path of /xyz,
RewriteEngine On
RewriteBase /xyz
RewriteRule ^oldstuff\.html$ newstuff.html
```

In Example 7–10, a request to `/xyz/oldstuff.html` gets correctly rewritten to the physical file `/abc/def/newstff.html`.

Rewrite Rules Hints

Table 7–3 provide hints for using rewrite rules.

Table 7–3 Rewrite Rules Hints

Value	Definition
.	Any single character
[char]	Any character listed within a square bracket

Table 7–3 (Cont.) Rewrite Rules Hints

Value	Definition
b*	Any character b any number of times
.*	Any character any number of times

For example, if you want to redirect requests from `/demo1`, `/demo2`, and `/demo3` to `/alldemos`, write the rewrite rule as one of the following:

```
RewriteRule /demo. /alldemos [R]
```

or,

```
RewriteRule /demo [123] /alldemos [R]
```

If you intend that `/DemoA`, `/DemoB`, and `/DemoC` to be redirected to `/alldemos`, add NC (no case) to the preceding rewrite rules, such as:

```
RewriteRule /demo [123] /alldemos [R, NC]
```

This rewrite rule will not work to redirect from `/demonstration1` to `/demos`, because "." works form one character only. To enable redirection of all URLs beginning with "demo", irrespective of subsequent characters, use the rewrite rule as follows:

```
RewriteRule ^/demo* /alldemos [R, NC]
```

In the preceding example, ^ means the beginning, * means any character after demo.

If there was a request for `/demo1/not_just_index.html`, al the preceding rewrite rules would have redirected the request the request to `/alldemos/index.html`, that may not be what you want. It is quite possible that you may want to redirect to the corresponding files in `/alldemos`, as listed in Table 7–4.

Table 7–4 Request Redirection

Request for	Redirected to
`/demo1/happy.html`	`/alldemos/happy.html`
`/demo1/go.jpg`	`/alldemos/go.jpg`
`/demos1/lucky.jpg`	`/alldemos/lucky.jpg`

Then you have to use substitution in your rewrite rule as follows:

```
RewriteRule ^/demos1(.*)$ //alldemos/$1 [R NC]
```

The explanation for this rule is:

Take the value of the expression, such as `happy.html`, `go.jpg`, and `lucky.jpg`, that appears after `demo1` as variables ($1) and substitute it after `/alldemos/`.

Redirection Examples

For redirecting requests from the `DocumentRoot` to a directory called `newroot`, set the following `mod_rewrite` directives:

```
RewriteEngine On
RewriteRule ^/(.*)$ /newroot/$1 [R]
```

For directing requested for files from one directory (`olddir`) to another (`newdir`), set the following directives:

```
RewriteEngine On
RewriteRule ^/olddir(.*)$ /newdir/$1 [R]
```

In each of these cases, you should ensure that the requested resources are indeed available in the redirected location. The mod_rewrite module does not ensure the existence of the requested resource in the new location.

For disabling all requests using the HTTP TRACE method, set the following mod_rewrite directives:

```
RewriteEngine On
RewriteCond %{REQUEST_METHOD} ^TRACE
RewriteRule .* - [F]
```

See Also: Module mod_rewrite in the Apache Server documentation.

mod_setenvif

Enables you to set environment variables based on characteristics of a request.

See Also: Module mod_setenvif in the Apache Server documentation.

mod_speling

Attempts to correct misspelled or miscapitalized URLs.

See Also: Module mod_speling in the Apache Server documentation.

mod_status

Displays an HTML page of server activity and performance.

See Also: Module mod_status in the Apache Server documentation.

mod_unique_id

Creates a unique ID for each request. This module is available on UNIX only.

See Also: Module mod_unique_id in the Apache Server documentation.

mod_userdir

Maps requests to user-specific directories.

See Also: Module mod_userdir in the Apache Server documentation.

mod_usertrack

Tracks user activity by creating a log.

See Also: Module `mod_usertrack` in the Apache Server documentation.

mod_vhost_alias

Enables dynamically configured mass virtual hosting.

See Also: Module `mod_vhost_alias` in the Apache Server documentation.

mod_wchandshake

Provides automatic discovery of Oracle HTTP Server by OracleAS Web Cache. If OracleAS Web Cache is not used, this module can be disabled. It is an Oracle module.

8 Managing Security

This chapter contains an overview of Oracle HTTP Server security features, and provides configuration information for setting up a secure Web site.

Topics discussed are:

- About Oracle HTTP Server Security
- Classes of Users and Their Privileges
- Resources Protected
- Authentication and Authorization Enforcement
- Understanding Port Tunneling
- Leveraging Oracle Identity Management Infrastructure

> **See Also:** For additional information about security, refer to the following documents:
>
> - The *Oracle Application Server Security Guide* provides an overview of Oracle Application Server security and its core functionality.
> - The *Oracle Identity Management Concepts and Deployment Planning Guide* provides guidance for administrators of the Oracle security infrastructure.

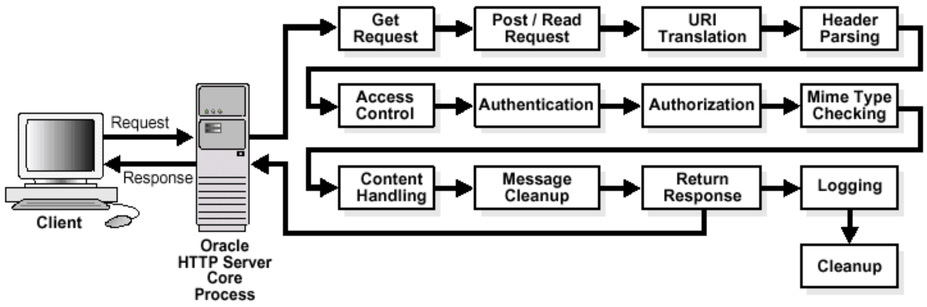
About Oracle HTTP Server Security

Security can be organized into the three categories of authentication, authorization, and confidentiality. Oracle HTTP Server provides support for all three of these categories. It is based on the Apache Web server, and its security infrastructure is primarily provided by the Apache modules, mod_auth and mod_access, and the Oracle modules, mod_ossl and mod_osso. `mod_auth` provides **authentication** based on user name and password pairs, `mod_access` controls access to the server based on the characteristics of a request, such as hostname or IP address, `mod_ossl` provides confidentiality and authentication with X.509 client certificates over SSL, and `mod_osso` enables **single sign-on** authentication for Web applications.

Based on the Apache model, Oracle HTTP Server provides access control, authentication, and authorization methods that can be configured with access control directives in the httpd.conf file. When URL requests arrive at Oracle HTTP Server, they are processed in a sequence of steps determined by server defaults and configuration parameters. The steps for handling URL requests are implemented through a module or plug-in architecture that is common to many Web listeners.

Figure 8–1 shows how URL requests are handled by the server. Each step in this process is handled by a server module depending on how the server is configured. For example, if basic authentication is used, then the steps labeled "Authentication" and "Authorization" in Figure 8–1 represent the processing of the mod_auth module.

Figure 8–1 Steps for Handling URL Requests in Oracle HTTP Server



Classes of Users and Their Privileges

Oracle HTTP Server authorizes and authenticates users before allowing them to access, or modify resources on the server. The following are three classes of users that access the server using Oracle HTTP Server, and their privileges:

- Users that access the server without providing any authentication. They have access to unprotected resources only.
- Users that have been authenticated and potentially authorized by modules within Oracle HTTP Server. This includes users authenticated by mod_auth and mod_ossl. Such users have access to URLs defined in http.conf file.

  See Also: "Authentication and Authorization Enforcement" on page 8-3

- Users that have been authenticated through mod_osso and Single Sign-On server. These users have access to resources allowed by Single Sign-On.

  See Also: *Oracle Application Server Single Sign-On Administrator's Guide*

Resources Protected

Oracle HTTP Server is configured to protect resources such as:

- Static content such as static HTML pages, graphics interchange format, .gif, files, and other static files that Oracle HTTP Server provides directly.
- CGI/FastCGI scripts, simple scripts or programs that Oracle HTTP Server invokes directly.
- Content generated by modules within Oracle HTTP Server. Modules such as mod_perl generate responses that are returned to the client.
- Oracle Application Server components that exist behind Oracle HTTP Server, including servlets and JSPs running with OC4J that are accessed through mod_

oc4j. Oracle HTTP Server forms the first line of authentication and authorization for these components, although further authentication may occur at the component level.

Authentication and Authorization Enforcement

Oracle HTTP Server provides user authentication and authorization at two stages:

- Host-based Access Control (stage one): This is based on the details of the incoming HTTP request and its headers, such as IP addresses or host names.
- User Authentication and Authorization (stage two): This is based on different criteria depending on the HTTP server configuration. The server can be configured to authenticate users with user name and password pairs that are checked against a list of known users and passwords. You can also configure the server to use single sign-on authentication for Web applications or X.509 client certificates over SSL.

Host-based Access Control

Early in the request processing cycle, access control is applied, which can inhibit further processing based on the host name, IP address, or other characteristics such as browser type. You use the `deny`, `allow`, and `order` directives to set this type of access control. These restrictions are configured with Oracle HTTP Server configuration directives and can be based on particular files, directories, or URL formats using the <Files>, <Directory>, and <Location> container directives as shown in the Example 8–1:

Example 8–1 Host-based Access Control

```
<Directory /internalonly/>
  order deny, allow
  deny from all
  allow from 192.168.1.* us.oracle.com
</Directory>
```

In Example 8–1, the `order` directive determines the order in which Oracle HTTP Server reads the conditions of the `deny` and `allow` directives. The `deny` directive ensures that all requests are denied access. Then, using the `allow` directive, requests originating from any IP address in the 192.168.1.* range, or with the domain name `us.oracle.com` are allowed access to files in the directory `/internalonly/`. It is common practice to specify both `allow` and `deny` in host-based authentication to make the access policy explicit.

If you want to match objects at the file system level, then you must use `<Directory>` or `<Files>`. If you want to match objects at the URL level, then you must use `<Location>`.

> **Note:** Allowing or restricting access based on a host name for Internet access is not considered a very good method of providing security, because host names are easy to spoof. While the same is true of IP addresses, sabotage is more difficult. However, setting access control with intranet IP address ranges is reasonable because the same risks do not apply. This assumes that your firewalls have been properly configured.

Access Control for Virtual Hosts

To set up access control for virtual hosts, place the Include directive inside a virtual host container in the server configuration file, httpd.conf. When used in a virtual host container, the Include directive specifies an access control policy contained in a file. Example 8–2 shows an excerpt from an httpd.conf file which provides the syntax for using Include this way:

Example 8–2 Using AccessConfig to Set Up Access Control

```
...
<VirtualHost ip_address_of_host.some_domain.com>
  ... virtual host directives ...
  Include conf/include.conf
</VirtualHost>
```

Using mod_access and mod_setenvif for Host-based Access Control

Using host-based access control schemes, you can control access to restricted areas based on where HTTP requests originate. Oracle HTTP Server uses mod_access and mod_setenvif to perform host-based access control. mod_access provides access control based on client hostname, IP address, or other characteristics of the client request, and mod_setenvif provides the ability to set environment variables based upon attributes of the request. When you enter configuration directives into the httpd.conf file that use these modules, the server fulfills or denies requests based on the address or name of the host, or based on the HTTP request header contents.

You can use host-based access control to protect static HTML pages, applications, or components.

Oracle HTTP Server supports four host-based access control schemes:

- Controlling Access by IP Address
- Controlling Access by Domain Name
- Controlling Access by Network or Netmask
- Controlling Access with Environment Variables

All of these allow you to specify the machines from which access to protected areas is granted or denied. Your decision to choose one or more of the host-based access control schemes is determined by which scheme most efficiently protects your restricted content and applications, or which scheme is easiest to maintain.

Controlling Access by IP Address Controlling access with IP addresses is a preferred method of host-based access control. It does not require DNS lookups that consume time, system resources, and make your server vulnerable to DNS spoofing attacks.

Example 8–3 Controlling Access by IP Address

```
<Directory /secure_only/>
  order deny,allow
  deny from all
  allow from 207.175.42.*
</Directory>
```

In Example 8–3, requests originating from all IP addresses except 207.175.42.* range are denied access to the /secure_only/ directory.

Controlling Access by Domain Name Domain name-based access control can be used with IP address-based access control to solve the problem of IP addresses changing without warning. When you combine these methods, if an IP address changes, then the secure areas of your site are still protected because the domain names you want to keep out will still be denied access.

To combine domain name-based with IP address-based access control, use the syntax shown in Example 8–4:

Example 8–4 controlling Access by Domain Name

```
<Directory /co_backgr/>
  order allow,deny
  allow from all
  # 141.217.24.* is the IP for malicious.cracker.com
  deny from malicious.cracker.com 141.217.24.*
</Directory>
```

In Example 8–4, all requests for directory `/co_backgr/` are accepted except those that originate from the domain name `malicious.cracker.com` or the IP address 141.217.24.* range. Although this is not a fool proof precaution against domain name or IP address spoofing, it protects your site from `malicious.cracker.com` even if they change their IP address.

Controlling Access by Network or Netmask You can control access based on subsets of networks, specified by IP address. The syntax is shown in Example 8–5:

Example 8–5 Controlling Access by Network or Netmask

```
<Directory /payroll/>
  order deny,allow
  deny from all
  allow from 10.1.0.0/255.255.0.0
</Directory>
```

In Example 8–5, access is allowed from a network/netmask pair. A netmask shows how an IP address is to be divided into network, subnet, and host identifiers. Netmasks enable you to refer to only the host ID portion of an IP address.

The netmask in Example 8–5, 255.255.0.0, is the default netmask setting for a Class B address. The binary ones (decimal 255) mask the network ID and the binary zeroes (decimal 0) retain the host ID of a given IP address.

Controlling Access with Environment Variables You can use arbitrary environment variables for access control, instead of using IP addresses or domain names. Use `BrowserMatch` and `SetEnvIf` directives for this type of access control.

> **Note:** Typically, `BrowserMatch` and `SetEnvIf` are not used to implement security policies. Instead they are used to provide different handling of requests based on browser types and versions.

Use `BrowserMatch` when you want to base access on the type of browser used to send a request. For instance, if you want to allow access only to requests that come from a Netscape browser, then use the syntax shown in Example 8–6:

Example 8–6 Controlling Access with Environment Variables

```
BrowserMatch ^Mozilla netscape_browser
<Directory /mozilla-area/>
  order deny,allow
  deny from all
  allow from env=netscape_browser
</Directory>
```

Use `SetEnvIf` when you want to base access on header information contained in the HTTP request. For instance, if you want to deny access from any browsers using HTTP version 1.0 or earlier, then use the syntax shown in Example 8–7:

Example 8–7 Controlling Access with SetEnv

```
SetEnvIf Request_Protocol ^HTTP/1.1 http_11_ok
<Directory /http1.1only/>
  order deny,allow
  deny from all
  allow from env=http_11_ok
</Directory>
```

User Authentication and Authorization

Basic authentication prompts for a user name and password before serving an HTTP request. When a browser requests a page from a protected area, Oracle HTTP Server responds with an unauthorized message (status code 401) containing a `WWW-Authenticate:` header and the name of the realm configured by the configuration directive, `AuthName`. When the browser receives this response, it prompts for a user name and password. After the user enters a user name and password combination, the browser sends this information back to the server in an Authorization header. In the authorization header message, the user name and password are encoded as a base 64 encoded string.

User authorization involves checking the authenticated user against an access control list that is associated with a specific server resource such as a file or directory. To configure user authorization, place the `require` directive in the `httpd.conf` file, usually within a virtual host container. User authorization is commonly used in combination with user authentication. After the server has authenticated a user's user name and password, then the server compares the user to an access control list associated with the requested server resource. If Oracle HTTP Server finds the user or the user's group on the list, then the resource is made available to that user.

Using mod_auth to Authenticate Users

User authentication is based on user names and passwords that are checked against a list of known users and passwords. These user name and password pairs may be stored in a variety of forms, such as a text file, database, or directory service. Then configuration directives are used in `httpd.conf` to configure this type of user authentication on the server. `mod_auth` uses the `AuthUserFile` directive to set up basic authentication. It supports only files.

Any authentication scheme that you devise requires that you use a combination of the configuration directives listed in Table 8–1.

Table 8–1 Directives Descriptions

Directive Name	Description
`AuthName`	Defines the name of the realm in which the user names and passwords are valid. Use quotation marks if the name includes spaces.
`AuthType`	Specifies the authentication type. Most authentication modules use basic authentication, which transmits user names and passwords in clear text. This is not recommended.
`AuthUserFile`	Specifies the path to a file that contains user names and passwords.
`AuthGroupFile`	Specifies the path to a file that contains group names and their members.

Using mod_osso to Authenticate Users

`mod_osso` enables single-sign on for Oracle HTTP Server. `mod_osso` examines incoming requests and determines whether the resource requested is protected, and if so, retrieves the Oracle HTTP Server cookie for the user.

Through `mod_osso`, Oracle HTTP Server becomes a single sign-on (SSO) partner application enabled to use SSO to authenticate users and obtain their identity using OracleAS Single Sign-On, and to make user identities available to Web applications as an Apache header variable.

Using `mod_osso`, Web applications can register URLs that require SSO authentication. When Oracle HTTP Server receives URL requests, `mod_osso` detects which requests require SSO authentication and redirects them to the SSO server. Once SSO server authenticates the users, it passes the user's authenticated identity back to `mod_osso` in a secure token, or cookie. `mod_osso` retrieves the user's identity from the cookie and propagates the user's identity information to applications running in Oracle HTTP Server instance. `mod_osso` can propagate the user's identity information to applications running in CGI, and those running in OC4J, and it can also authenticate users for access to static files.

See Also:

- *Oracle Application Server Single Sign-On Administrator's Guide*
- "Leveraging Oracle Identity Management Infrastructure" on page 8-14

Using mod_ossl to Authenticate Users

`mod_ossl` is a plug-in to Oracle HTTP Server that enables the server to use SSL. `mod_ossl` replaces `mod_ssl` in the Oracle HTTP Server distribution. Oracle no longer supports `mod_ssl`.

See Also: "Enabling SSL for Oracle HTTP Server" on page 9-1 for information on enabling and configuring SSL using `mod_ossl` directives.

Understanding Port Tunneling

Port tunneling allows all communication between Oracle HTTP Server and OC4J to happen on a single, or a small number of ports. Previously, the firewall configuration had to include port information for many ports to handle communication between Oracle HTTP Server and multiple OC4J instances.

> **See Also:** *Oracle HTTP Server Administrator's Guide* for a detailed description of port tunneling.

Configuring Port Tunneling

Perform the following three tasks to configure port tunneling:

- Task 1: Configure opmn.xml
- Task 2: Configure iaspt.conf
- Task 3: Configure mod_oc4j.conf

Task 1: Configure opmn.xml

Perform the following steps on the middle-tier install (and not the standalone Oracle HTTP Server 2.0 installation) to start one or more iaspt daemons:

1. By default, there is an opmn.xml entry for iaspt that is disabled. Enable iaspt by editing opmn.xml and changing status="disable" to status="enable".
2. Optionally, you may also change the TCP/IP ports used by the iaspt daemon by changing the port "range" and the number of the iaspt daemon processes by changing "numprocs".

   The following is a complete example configuration for the iaspt daemon. It contains all possible configuration elements/attributes that can be used with this component.

   ```
   <module path="/ORACLE_HOME/opmn/lib/libopmniaspt">
     <module-id id="IASPT" />
   </module>
   <ias-component id="IASPT" status="enabled" id-matching="false">
     <process-type id="IASPT" module-id="IASPT">
       <port id="ajp" range="6701-6703"/>
       <process-set id="IASPT" restart-on-death="true" numprocs="3"/>
     </process-type>
   </ias-component>
   ```

3. Run the following command to direct the opmn daemon to reload its configuration file:

   ```
   opmnctl reload
   ```

Task 2: Configure iaspt.conf

Perform the following steps on the middle-tier install (and not the standalone Oracle HTTP Server 2.0 installation) to configure iaspt.conf to specify an SSL wallet for the iaspt daemon(s) to use:

1. Communication between mod_oc4j and iaspt is always encrypted, therefore an SSL wallet file must be configured for the iaspt daemon(s). By default, this wallet is the same as the Oracle HTTP Server wallet. You may change the default by editing the following values in iaspt.conf:

   ```
   wallet-file=<path to wallet file>
   wallet-password=<password>
   ```

See Also:

- "wallet-file" on page 8-12
- "wallet-password" on page 8-13

2. Start the `iaspt` daemon(s) using the following command:

```
opmnctl startall
```

Task 3: Configure mod_oc4j.conf

Perform the following steps on the standalone Oracle HTTP Server 2.0 installation (and not the middle tier install) to configure `mod_oc4j.conf` to route requests using `iaspt`:

1. Enable port tunneling by adding the following line in `mod_oc4j.conf`:

```
Oc4jiASPTActive on
```

See Also: "Oc4jiASPTActive" on page 8-11

2. Specify an SSL wallet and wallet password for `mod_oc4j.conf` by adding the following two lines in `mod_oc4j.conf`:

```
Oc4jiASPTWalletFile <path to wallet file>
Oc4jiASPTWalletPassword <password of wallet>
```

This wallet may be the same as used by Oracle HTTP Server and/or `iaspt`.

See Also:

- "Oc4jiASPTWalletFile" on page 8-11
- "Oc4jiASPTWalletPassword" on page 8-12
- *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.

3. Specify the host and port addresses of the `iaspt` daemons. For example, add the following line to `mod_oc4j.conf`:

```
Oc4jiASPTProcess myhost.us.oracle.com:6701
```

You may add as many `Oc4jiASPTProcess` lines as you have `iaspt` daemons. The host and port addresses must match those of your configured `iaspt` daemons. For example, to route requests to the three `iaspt` daemons configured in the example in step 2 of "Task 1: Configure opmn.xml" on page 8-8, add the following three lines:

```
Oc4jiASPTProcess myhost.us.oracle.com:6701
Oc4jiASPTProcess myhost.us.oracle.com:6702
Oc4jiASPTProcess myhost.us.oracle.com:6703
```

See Also: "Oc4jiASPTProcess" on page 8-11

4. Restart Oracle HTTP Server for the changes to take effect, using the following command:

- UNIX: `ORACLE_HOME/opmn/bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`
- Windows: `ORACLE_HOME\opmn\bin> opmnctl [verbose] restartproc ias-component=HTTP_Server`

Configuring SSL for Port Tunneling

This section contains information about configuring SSL between `iaspt` and OC4J

By default, the `iaspt` daemons and the OC4J processes communicate with unencrypted data. Perform the following steps to configure SSL communication between these processes:

1. In `iaspt.conf`, change the value `"destination-ssl"` from `"false"` to `"true"`.
2. To configure the OC4J process to use SSL, refer to the *Oracle Application Server Containers for J2EE Security Guide*.

Port Tunneling Configuration Reference

This section contains information about the following configuration files and their parameters:

- opmn.xml
- mod_oc4j.conf
- iaspt.conf

opmn.xml

Describes the process that OPMN manages within an Oracle Application Server installation.

> **See Also:** "opmn.xml" on page B-3

As part of port tunneling, an **entry** that describes the `iaspt` daemon process to be started should exist in OPMN. This entry describes the following:

- number of `iaspt` daemon processes to start.
- ports that these processes can use.

> **See Also:** iaspt.conf on page 8-12

An out of the box Oracle Application Server installation contains an `iaspt` component in `opmn.xml`, but it is disabled by default.

mod_oc4j.conf

Configures `mod_oc4j` with Oracle HTTP Server.

> **See Also:** "mod_oc4j.conf" on page B-3

For port tunneling, you need to add directives that specify the following:

- whether port tunneling should be used
- static location for an `iaspt` daemon process

- location of SSL certificates to be used in establishing connections with the iaspt daemon processes.

See Also: iaspt.conf on page 8-12

By default, mod_oc4j communicates directly to OC4J. For port tunneling process, mod_oc4j should communicate to OC4J through the iaspt daemon.

The following directives used to connect mod_oc4j to the iaspt daemon:

- Oc4jiASPTActive
- Oc4jiASPTProcess

Oc4jiASPTActive Indicates whether mod_oc4j needs to consider port tunneling when routing requests. This should not be configured to "On" if Oc4jEnableSSL is configured to "On". To enable port tunneling process, set this directive to "On".

Category	Value
Parameter Name	Oc4jiASPTActive
Parameter Type	string
Valid Values	On/Off
Default Value	Off

Oc4jiASPTProcess Describes the listening host and port of a port tunneling process. There can be multiple instances of this directive within a mod_oc4j.conf file for multiple port tunneling processes.

The syntax for this directive is host:port. The host value should be the hostname of a machine where an iaspt daemon is running. The port value should match the port configured in opmn.xml for that iaspt. Both regular hostname and IP address are allowed for host.

Category	Value
Parameter Name	Oc4jiASPTProcess
Parameter Type	string
Valid Values	host:port values of the available iaspt daemons.
Default Value	N/A
Syntax	host:port For example: *myhost.us.oracle.com*:6667

mod_oc4j should use SSL when communicating with the iaspt daemon. The following are the directives used to enable SSL:

- Oc4jiASPTWalletFile
- Oc4jiASPTWalletPassword

Oc4jiASPTWalletFile Specifies the location of an Oracle Wallet file that contains SSL certificates that are used for SSL communication with the iaspt daemon.

Category	Value
Parameter Name	`Oc4jiASPTWalletFile`
Parameter Type	string
Valid Values	Path to a wallet file that contains the SSL certificate to be used when establishing SSL connections to the iaspt daemon.
Default Value	N/A
Syntax	Valid filename For example: `/foo/bar/`*`myfilename`*

Oc4jiASPTWalletPassword Specifies the value of the obfuscated password used for authentication when opening the wallet file. This value is obtained using the utility provided with Oracle Wallet Manager.

Category	Value
Parameter Name	`Oc4jiASPTWalletPassword`
Parameter Type:	string
Valid Values	Password used for authentication when opening the wallet file specified by Oc4jiASPTWalletFile.
Default Value	N/A

> **See Also:** *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.

iaspt.conf

Configures port tunneling.

> **See Also:** "iaspt.conf" on page B-2

It specifies the following information:

- Wallet file and password that should be used
- Log file location and log level
- Port that `iaspt` daemon should listen on (optionally). This port can either be specified in `iaspt.conf`, or can be passed in from `opmn.xml` by specifying a range of ports. By doing so, more than one port tunneling process can use the same `iaspt.conf` file.

The `iaspt.conf` file is a set of name value pairs. The following are the names of the parameters accepted:

- wallet-file
- wallet-password
- log-file
- log-level
- iaspt-port

wallet-file Specifies the location of an Oracle Wallet file that contains SSL certificates that are used for SSL communication with peers.

Category	Value
Parameter Name	`wallet-file`
Parameter Type	string
Valid Values	Path to a wallet file that contains the SSL certificate to be used when establishing SSL connections to other processes.
Default Value	N/A
Syntax	Valid filename For example: `/foo/bar/`*`myfilename`*

wallet-password Specifies the value of the password used for authentication when opening the wallet file. This value is obtained using the utility provided with Oracle Wallet Manager.

Category	Value
Parameter Name	`wallet-password`
Parameter Type	string
Valid Values	Password used for authentication when opening the wallet file specified by wallet-file
Default Value	N/A

See Also: *Oracle Application Server Administrator's Guide* for information on Oracle Wallet Manager.

log-file Specifies the path to a log file where `iaspt` daemon logging messages are written to.

Category	Value
Parameter Name	`log-file`
Parameter Type	string
Valid Values	Path to a log file where `iaspt` daemon logging messages are written to.
Default Value	N/A
Syntax	Valid filename For example: `/foo/bar/`*`myfilename`*

log-level Specifies the logging level where 9 is the highest and 0 implies no logging.

Category	Value
Parameter Name	`log-level`
Parameter Type	integer
Valid Values	Integer from 0 to 9
Default Value	3

iaspt-port Specifies the port value that the iaspt daemon should accept connections on. This is optional.

Category	Value
Parameter Name	`iaspt-port`
Parameter Type	integer
Valid Values	Valid TCP/IP port value
Syntax	Integer For example: 9898
Default Value	N/A

Leveraging Oracle Identity Management Infrastructure

This section discusses how Oracle HTTP Server uses the Oracle Identity Management Infrastructure.

Overview

Oracle Identity Management is an integrated infrastructure that the Oracle Application Server relies on for distributed security. It consists of Oracle Internet Directory, Oracle Directory Integration and Provisioning, Delegated Administrative Service, OracleAS Single Sign-On, and Oracle Certificate Authority.

> **See Also:** *Oracle Identity Management Concepts and Deployment Planning Guide*

Using Oracle Application Server Single Sign-On and mod_osso

Oracle Application Server supports single sign-on (SSO) to Web-based applications through OracleAS Single Sign-On. OracleAS Single Sign-On enables you to log in to Oracle Application Server and gain access to those applications for which you have authorization, without requiring to re-enter a user name and password for each application. It is fully integrated with Oracle Internet Directory, which stores user information. It supports LDAP-based user and password management through Oracle Internet Directory.

`mod_osso`, an Oracle HTTP Server module, enables the transparent use of OracleAS Single Sign-On across all of Oracle Application Server. Through `mod_osso`, Oracle HTTP Server becomes a SSO partner application enabled to use SSO to authenticate users and obtain their identity, and to make user identities available to Web applications as an Apache header variable.

> **See Also:** "Using mod_osso to Authenticate Users" on page 8-7

9

Enabling SSL for Oracle HTTP Server

This chapter contains information about enabling and configuring SSL for Oracle HTTP Server. Topics discussed are:

- Overview
- Configuring SSL
- Additional SSL Features
- Using SSL Configuration Directives

Overview

Secure Sockets Layer (SSL) is an encrypted communication protocol that is designed to securely send messages across the Internet. It resides between Oracle HTTP Server on the application layer and the TCP/IP layer, transparently handling encryption and decryption when a secure connection is made by a client.

One common use of SSL is to secure Web HTTP communication between a browser and a Web server. This case does not preclude the use of non-secured HTTP. The secure version is simply HTTP over SSL (named HTTPS). The differences are that HTTPS uses the URL scheme `https://` rather than `http://`, and its default communication port is 4443 on UNIX or 443 on Windows.

`mod_ossl` is a plug-in to Oracle HTTP Server that enables the server to use SSL.

See Also: *Oracle Application Server Administrator's Guide*

Configuring SSL

By default, SSL is disabled when you install Oracle Application Server. Perform these tasks to enable and configure SSL:

- Task 1: Creating a Real Wallet
- Task 2: Enabling SSL
- Task 3: (Optional) Customizing Your Configuration

Task 1: Creating a Real Wallet

To configure Oracle HTTP Server for SSL, you need a **wallet** that contains the certificate for the server. Wallets store your credentials, such as certificate requests, certificates, and private keys.

The default wallet that is automatically installed with Oracle HTTP Server is for testing purposes only. A real wallet has to be created for your production server. The default wallet is located in `ORACLE_HOME//ohs/conf/ssl.wlt/default`. You can either place the new wallet in that location, or change the SSLWallet directive in `ORACLE_HOME/ohs/conf/ssl.conf` to point to the location of your real wallet.

> **See Also:** *Oracle Application Server Administrator's Guide* for instructions on creating a wallet. It is important that you do the following:
>
> 1. Generate a certificate request: For the Common Name, specify the name or alias of the site you are configuring.
> 2. Set the auto-login feature for your wallet: Make sure that you enable this auto-login feature. The default wallet has this feature disabled.

Task 2: Enabling SSL

Perform the following steps to enable SSL manually:

1. Open opmn.xml in a text editor.
2. In the `<ias-component id="HTTP_Server">` entry, change the start mode from "ssl-disabled" to "ssl-enabled". After modification is made, the entry should look like the following:

   ```
   <data id="start-mode" value="ssl-enabled"/>
   ```

3. Save and close `opmn.xml`.
4. Reload OPMN using the following command:

   ```
   opmnctl reload
   ```

5. Stop Oracle HTTP Server using the following command:
   - UNIX: `ORACLE_HOME/opmn/bin> opmnctl [verbose] stopproc ias-component=HTTP_Server`
   - Windows: `ORACLE_HOME\opmn\bin> opmnctl [verbose] stopproc ias-component=HTTP_Server`
6. Start Oracle HTTP Server using the following command:
   - UNIX: `ORACLE_HOME/opmn/bin> opmnctl [verbose] startproc ias-component=HTTP_Server`
   - Windows: `ORACLE_HOME\opmn\bin> opmnctl [verbose] startproc ias-component=HTTP_Server`

   > **Note:** Be sure that you stop and start Oracle HTTP Server as per the instructions. Restarting Oracle HTTP Server does not yield the same result as stopping and starting it.

7. You can verify if SSL was enabled successfully by navigating to the SSL port, for example:

   ```
   HTTPS://hostname:4443
   ```

Task 3: (Optional) Customizing Your Configuration

Optionally, you can further customize your configuration using `mod_ossl` directives.

> **See Also:** "Using mod_ossl Directives" on page 9-4

> **Note:** The templates files installed during installation contain all the necessary SSL configuration directives and a default setup for SSL.

To enable client authentication, do the following:

1. Specify SSLVerifyClient on the server side.
2. Use proper client certificate on your client side for the HTTPS connection. Refer to your client documentation for information on getting and using a client certificate. Be sure that your client certificate is trusted by the server wallet.

> **See Also:** *Oracle Application Server Administrator's Guide* for instructions on how to import a trusted certificate into your wallet.

Additional SSL Features

This section contains SSL features that are supported for this release.

- Global Server ID Support
- PKCS #11 Support

Global Server ID Support

This feature adds support SSL protocol features called variously "step-up", "server gated crypto" or "global server ID". "Step-up" is a feature that allows old, weak encryption browsers, to "step-up" so that public keys greater than 512 bits and bulk encryption keys greater than 64 bits can be used in the SSL protocol. This means that server X.509 certificates that contain public keys in excess of 512 bits and which contain "step-up" digital rights can now be used by Oracle Application Server. Such certificates are often called "128 bit" certificates, even though the certificate itself typically contains a 1024 bit certificate. The Verisign Secure Site Pro is an example of such a certificate which can now be used by Oracle Application Server.

Global Server ID functionality is provided by default, there is no configuration necessary.

PKCS #11 Support

Public-Key Cryptography Standards #11, or PKCS #11 for short, is a public key cryptography specification that outlines how systems use hardware security modules, which are basically "boxes" where cryptographic functions (encryption/decryption) are performed and where encryption keys are stored.

Oracle HTTP Server supports the option of having dedicated SSL hardware through nCipher. nCipher is a certified third party accelerator that improves the performance of the PKI cryptography that SSL uses.

See Also:

- *Oracle Application Server Administrator's Guide*
- http://www.ncipher.com

Using SSL Configuration Directives

mod_ossl provides standard support for HTTPS protocol connections to Oracle Application Server. It enables secure connections between Oracle HTTP Server and a browser client by using an Oracle-provided encryption mechanism over SSL. It may also be used for authentication over the Internet through the use of digital certificate technology. It supports SSL v. 3.0, and provides:

- Encrypted communication between client and server, using **RSA** or **DES** encryption standards.
- Integrity checking of client/server communication using **MD5** or **SHA** checksum algorithms.
- Certificate management with Oracle **wallet**s.

The following mod_ssl directives are not supported by mod_ossl.

- SSLRandomSeed
- SSLCertificateFile
- SSLCertificateKeyFile
- SSLCertificateChainFile
- SSLCACertificateFile
- SSLCACertificatePath
- SSLVerifyDepth

Note: The server will not start if these directives are used.

Using mod_ossl Directives

To configure SSL for your Oracle HTTP Server, enter the mod_ossl directives you want to use in the httpd.conf file.

The following directives are described:

- SSLAccelerator
- SSLCARevocationFile
- SSLCARevocationPath
- SSLCipherSuite
- SSLEngine
- SSLLog
- SSLLogLevel
- SSLMutex
- SSLOptions
- SSLPassPhraseDialog

- SSLProtocol
- SSLRequire
- SSLRequireSSL
- SSLSessionCache
- SSLSessionCacheTimeout
- SSLVerifyClient
- SSLWallet
- SSLWalletPassword

SSLAccelerator

Specifies if SSL accelerator is used. Currently only nFast card is supported.

Category	Value
Valid Values	`yes/no`
Syntax	`SSLAccelerator yes|no`
Default	`SSLAccelerator no`
Context	server configuration

> **Note:** The `SSLAccelerator` directive has been deprecated. For information on enabling SSL acceleration support using a wallet, refer to the *Oracle Advanced Security Administrator's Guide* on `http://www.oracle.com/technology/documentation`.

SSLCARevocationFile

Specifies the file where you can assemble the Certificate Revocation Lists (CRLs) from **CA**s (Certificate Authorities) that you accept certificates from. These are used for client authentication. Such a file is the concatenation of various **PEM**-encoded CRL files in order of preference. This directive can be used alternatively or additionally to SSLCARevocationPath.

Category	Value
Syntax	`SSLCARevocationFile` *`file_name`*
Example	`SSLCARevocationFile /`*`ORACLE_HOME`*`/ohs/conf/ssl.crl/ca_bundle.crl`
Default	None
Context	server configuration, virtual host

SSLCARevocationPath

Specifies the directory where **PEM**-encoded Certificate Revocation Lists (CRLs) are stored. These CRLs come from the **CA**s (Certificate Authorities) that you accept certificates from. If a client attempts to authenticate itself with a certificate that is on one of these CRLs, then the certificate is revoked and the client cannot authenticate itself with your server.

Category	Value
Syntax	`SSLCARevocationPath` *`path/to/CRL_directory/`*
Example	`SSLCARevocationPath /`*`ORACLE_HOME`*`/ohs/conf/ssl.crl/`
Default	None
Context	server configuration, virtual host

SSLCipherSuite

Specifies the SSL **cipher suite** that the client can use during the SSL handshake. This directive uses a colon-separated cipher specification string to identify the cipher suite. Table 9–2 shows the tags you can use in the string to describe the cipher suite you want.

Tags are joined together with prefixes to form cipher specification string.

Category	Value
Valid Values	`none`: Adds the cipher to the list `+` : Adds the cipher to the list and place them in the correct location in the list `-` : Remove the cipher from the list (can be added later) `!` : Remove the cipher from the list permanently
Example	`SSLCipherSuite` *`ALL:!LOW:!DH`* In this example, all ciphers are specified except low strength ciphers and those using the **Diffie-Hellman key negotiation algorithm**.
Syntax	`SSLCipherSuite` *`cipher-spec`*
Default	`ALL:!ADH:!EXPORT56:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP`
Context	server configuration, virtual host, directory

Table 9–1 SSLCipher Suite Tags

Function	Tag	Meaning
Key exchange	`kRSA`	`RSA` key exchange
Key exchange	`kDHr`	Diffie-Hellman key exchange with RSA key
Authentication	`aNULL`	No authentication
Authentication	`aRSA`	`RSA` authentication
Authentication	`aDH`	Diffie-Hellman authentication
Encryption	`eNULL`	No encryption
Encryption	`DES`	`DES` encoding
Encryption	`3DES`	Triple `DES` encoding
Encryption	`RC4`	`RC4` encoding
Data Integrity	`MD5`	`MD5` hash function
Data Integrity	`SHA`	`SHA` hash function
Aliases	`SSLv3`	All SSL version 3.0 ciphers
Aliases	`EXP`	All export ciphers

Table 9–1 (Cont.) SSLCipher Suite Tags

Function	Tag	Meaning
Aliases	`EXP40`	All 40-bit export ciphers only
Aliases	`EXP56`	All 56-bit export ciphers only
Aliases	`LOW`	All low strength ciphers (export and single `DES`)
Aliases	`MEDIUM`	All ciphers with 128-bit encryption
Aliases	`HIGH`	All ciphers using triple `DES`
Aliases	`RSA`	All ciphers using `RSA` key exchange
Aliases	`DH`	All ciphers using Diffie-Hellman key exchange

Note: There are restrictions if export versions of browsers are used. Oracle module, `mod_ossl`, supports RC4-40 encryption only when the server uses 512 bit key size wallets.

Table 9–2 Cipher Suites Supported in Oracle Advanced Security 10i

Cipher Suite	Authentication	Encryption	Data Integrity
`SSL_RSA_WITH_3DES_EDE_CBC_SHA`	`RSA`	`3DES (168)`	`SHA`
`SSL_RSA_WITH_RC4_128_SHA`	`RSA`	`RC4 (128)`	`SHA`
`SSL_RSA_WITH_RC4_128_MD5`	`RSA`	`RC4 (128)`	`MD5`
`SSL_RSA_WITH_DES_CBC_SHA`	`RSA`	`DES (56)`	`SHA`
`SSL_DH_anon_WITH_3DES_EDE_CBC_SHA`	`DH anon`	`3DES (168)`	`SHA`
`SSL_DH_anon_WITH_RC4_128_MD5`	`DH anon`	`RC4 (128)`	`MD5`
`SSL_DH_anon_WITH_DES_CBC_SHA`	`DH anon`	`DES (56)`	`SHA`
`SSL_RSA_EXPORT_WITH_RC4_40_MD5`	`RSA`	`RC4 (40)`	`MD5`
`SSL_RSA_EXPORT_WITH_DES40_CBC_SHA`	`RSA`	`DES40 (40)`	`SHA`
`SSL_RSA_WITH_AES_128_CBC_SHA`	`RSA`	`AES (128)`	`SHA`
`SSL_RSA_WITH_AES_256_CBC_SHA`	`RSA`	`AES (256)`	`SHA`
`SSL_DHE_DSS_EXPORT_WITH_DES40_CBS_SHA`	`DH DSS`	`DES (40)`	`SHA`
`SSL_DHE_DSS_WITH_DES_CBC_SHA`	`DH DSS`	`DES (50)`	`SHA`
`SSL_DHE_DSS_WITH_3DES_EDE_CBC_SHA`	`DH DSS`	`3DES (168)`	`SHA`
`SSL_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA`	`DH RSA`	`DES (40)`	`SHA`
`SSL_DHE_RSA_WITH_DES_CBC_SHA`	`DH RSA`	`DES (56)`	`SHA`
`SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA`	`DH RSA`	`3DES (168)`	`SHA`
`SSL_DHE_DSS_EXPORT1024_WITH_DES_CBC_SHA`	`DH DSS`	`DES (40)`	`SHA`
`SSL_DHE_DSS_WITH_RC4_128_SHA`	`DH DSS`	`RC4 (128)`	`SHA`
`SSL_DHE_DSS_EXPORT1024_WITH_RC4_56_SHA`	`DH DSS`	`RC4 (56)`	`SHA`

SSLEngine

Toggles the usage of the SSL Protocol Engine. This is usually used inside a `<VirtualHost>` section to enable SSL for a particular virtual host. By default, the SSL Protocol Engine is disabled for both the main server and all configured virtual hosts. Example 9–1 is an example for using SSLEngine directive. The default SSL is 4443 on UNIX and 443 on Windows.

Example 9–1 Using SSLEngine Directive

```
<VirtualHost_dafault_:4443>
  SSLEngine on
  ...
</VirtualHost>
```

Category	**Value**
Syntax	`SSLEngine on|off`
Default	`SSLEngine off`
Context	server configuration, virtual host

SSLLog

Specifies where the SSL engine log file will be written. (Error messages will also be duplicated to the standard Oracle HTTP Server log file specified by the ErrorLog directive.)

Place this file at a location where only root can write, so that it cannot be used for symlink attacks. If the filename does not begin with a slash (/), it is assumed to be relative to the ServerRoot. If the filename begins with a bar (|), then the string following the bar is expected to be a path to an executable program to which a reliable pipe can be established.

This directive should occur only once per virtual server configuration.

Category	**Value**
Syntax	`SSLVerifyClient` *`path/to/filename`*
Default	None
Context	server configuration, virtual host

SSLLogLevel

Specifies the verbosity degree of the SSL engine log file.

Category	Value
Valid Values	The levels are (in ascending order, where each level is included in the levels preceding it): ■ `none`: No dedicated SSL logging is done. Messages of type 'error' are duplicated to the standard HTTP server log file specified by the `ErrorLog` directive. ■ `error`: Only messages of the type 'error' (conditions that stop processing) are logged. ■ `warn`: Messages that notify of non-fatal problems (conditions that do not stop processing) are logged. ■ `info`: Messages that summarize major processing actions are logged. ■ `trace`: Messages that summarize minor processing actions are logged. ■ `debug`: Messages that summarize development and low-level I/O operations are logged.
Syntax	`SSLLogLevel` *`level`*
Default	None
Context	server configuration, virtual host

SSLMutex

Type of semaphore (lock) for SSL engine's mutual exclusion of operations that have to be synchronized between Oracle HTTP Server processes.

Category	Value
Valid Values	■ `none`: Uses no mutex at all. Not recommended, because the mutex synchronizes the write access to the SSL session cache. If you do not configure a mutex, the session cache can become garbled. ■ `file:`path/to/mutex: Uses a file for locking. The process ID (PID) of the Oracle HTTP Server parent process is appended to the filename to ensure uniqueness. If the filename does not begin with a slash (/), it is assumed to be relative to `ServerRoot`. This setting is not available on Windows. ■ `sem`: Uses an operating system semaphore to synchronize writes. On UNIX, it would be a Sys V IPC semaphore; on Windows, it is a Windows Mutex. This is the best choice, if the operating system supports it.
Example	`SSLMutex file`*`:/usr/local/apache/logs/ssl_mutex`*
Syntax	`SSLMutex` *`type`*
Default	`SSLMutex` *`none`*
Context	server configuration

SSLOptions

Controls various runtime options on a per-directory basis. In general, if multiple options apply to a directory, the most comprehensive option is applied (options are not merged). However, if all of the options in an `SSLOptions` directive are preceded by a plus ('+') or minus ('-') symbol, then the options are merged. Options preceded by a plus are added to the options currently in force, and options preceded by a minus are removed from the options currently in force.

Category	Value
Valid Values	■ StdEnvVars: Creates the standard set of CGI/SSI environment variables that are related to SSL. This is disabled by default because the extraction operation uses a lot of CPU time and usually has no application when serving static content. Typically, you only enable this for CGI/SSI requests. ■ ExportCertData: Enables the following additional CGI/SSI variables: SSL_SERVER_CERT SSL_CLIENT_CERT SSL_CLIENT_CERT_CHAIN_n (where n= 0, 1, 2...) These variables contain the Privacy Enhanced Mail (**PEM**)-encoded X.509 certificates for the server and the client for the current HTTPS connection, and can be used by CGI scripts for deeper certificate checking. All other certificates of the client certificate chain are provided. This option is "Off" by default because there is a performance cost associated with using it. SSL_CLIENT_CERT_CHAIN_n variables are in the following order: SSL_CLIENT_CERT_CHAIN_0 is the intermediate CA who signs SSL_CLIENT_CERT. SSL_CLIENT_CERT_CHAIN_1 is the intermediate CA who signs SSL_CLIENT_CERT_CHAIN_0, and so forth, with SSL_CLIENT_ROOT_CERT as the root CA. ■ FakeBasicAuth: Translates the subject **distinguished name** of the client **X.509** certificate into an HTTP basic authorization user name. This means that the standard HTTP server authentication methods can be used for access control. Note that no password is obtained from the user; the string 'password' is substituted.
Valid Values (for SSLOptions continued)	■ StrictRequire: Denies access when, according to SSLRequireSSL or directives, access should be forbidden. Without StrictRequire, it is possible for a 'Satisfy any' directive setting to override the SSLRequire or SSLRequireSSL directive, allowing access if the client passes the host restriction or supplies a valid user name and password. Thus, the combination of SSLRequireSSL or SSLRequire with SSLOptions +StrictRequire gives mod_ossl the ability to override a 'Satisfy any' directive in all cases. ■ CompatEnvVars: Exports obsolete environment variables for backward compatibility to Apache SSL 1.x, mod_ssl 2.0.x, Sioux 1.0, and Stronghold 2.x. Use this to provide compatibility to existing CGI scripts. ■ OptRenegotiate: This enables optimized SSL connection renegotiation handling when SSL directives are used in a per-directory context.
Syntax	SSLOptions *[+-] option*
Default	None
Context	server configuration, virtual host, directory

SSLPassPhraseDialog

Type of pass phrase dialog for wallet access. mod_ossl asks the administrator for a pass phrase in order to access the wallet.

Category	Value
Valid Values	■ `builtin`: when the server is started, `mod_ossl` prompts for a password for each wallet. This cannot be used when Oracle HTTP Server is managed by OPMN. No user interaction is allowed when Oracle HTTP Server is started by OPMN. ■ `exec`:path/to/program - when the server is started, `mod_ossl` calls an external program configured for each wallet. This program is invoked with two arguments: servername:portnumber and `RSA` or `DSA`.
Syntax	`SSLPassPhraseDialog` *type*
Example	`SSLPassPhraseDialog exec:`*`/usr/local/apache/sbin/pfilter`*
Default	`SSLPassPhraseDialog` *`builtin`*
Context	server configuration

SSLProtocol

Specifies SSL protocol(s) for `mod_ossl` to use when establishing the server environment. Clients can only connect with one of the specified protocols.

Category	Value
Valid Values	SSLv2, SSLv3, TLSv1, ALL
Example	To specify only SSL version 3.0, set this directive to the following: `SSLProtocol` *`+SSLv3`*
Syntax	`SSLProtocol` *`[+-] protocol`*
Default	`SSLProtocol ALL`
Context	server configuration, virtual host

SSLRequire

Denies access unless an arbitrarily complex boolean expression is true.

Category	Value
Syntax	`SSLRequire` *`expression`*
Default	None
Context	directory

The *expression* must match the following syntax (given as a BNF grammar notation):

```
expr ::= "true" | "false"
"!" expr
expr "&&" expr
expr "||" expr
"(" expr ")"

comp ::=word "==" word | word "eq" word
word "!=" word |word "ne" word
word "<" word |word "lt" word
word "<=" word |word "le" word
word ">" word |word "gt" word
word ">=" word |word "ge" word
word "=~" regex
word "!~" regex
wordlist ::= word
wordlist "," word

word ::= digit
cstring
variable
function

digit ::= [0-9]+

cstring ::= "..."

variable ::= "%{varname}"
```

Table 9–3 and Table 9–4 list standard and SSL variables. These are valid values for varname.

```
function ::= funcname "(" funcargs ")"
```

For `funcname`, the following function is available:

```
file(filename)
```

The file function takes one string argument, the filename, and expands to the contents of the file. This is useful for evaluating the file's contents against a regular expression.

Table 9–3 lists the standard variables for SSLRequire varname.

Table 9–3 Standard Variables for SSLRequire Varname

Standard Variables	Standard Variables	Standard Variables
HTTP_USER_AGENT	PATH_INFO	AUTH_TYPE
HTTP_REFERER	QUERY_STRING	SERVER_SOFTWARE
HTTP_COOKIE	REMOTE_HOST	API_VERSION
HTTP_FORWARDED	REMOTE_IDENT	TIME_YEAR
HTTP_HOST	IS_SUBREQ	TIME_MON
HTTP_PROXY_CONNECTION	DOCUMENT_ROOT	TIME_DAY
HTTP_ACCEPT	SERVER_ADMIN	TIME_HOUR
HTTP:headername	SERVER_NAME	TIME_MIN
THE_REQUEST	SERVER_PORT	TIME_SEC
REQUEST_METHOD	SERVER_PROTOCOL	TIME_WDAY
REQUEST_SCHEME	REMOTE_ADDR	TIME
REQUEST_URI	REMOTE_USER	ENV:variablename
REQUEST_FILENAME		

Table 9–4 lists the SSL variables for SSLRequire varname.

Table 9–4 SSL Variables for SSLRequire Varname

SSL Variables	SSL Variables	SSL Variables
HTTPS	SSL_PROTOCOL	SSL_CIPHER_ALGKEYSIZE
SSL_CIPHER	SSL_CIPHER_EXPORT	SSL_VERSION_INTERFACE
SSL_CIPHER_USEKEYSIZE	SSL_VERSION_LIBRARY	SSL_SESSION_ID

Table 9–4 (Cont.) SSL Variables for SSLRequire Varname

SSL Variables	SSL Variables	SSL Variables
SSL_CLIENT_V_END	SSL_CLIENT_M_SERIAL	SSL_CLIENT_V_START
SSL_CLIENT_S_DN_ST	SSL_CLIENT_S_DN	SSL_CLIENT_S_DN_C
SSL_CLIENT_S_DN_CN	SSL_CLIENT_S_DN_O	SSL_CLIENT_S_DN_OU
SSL_CLIENT_S_DN_G	SSL_CLIENT_S_DN_T	SSL_CLIENT_S_DN_I
SSL_CLIENT_S_DN_UID	SSL_CLIENT_S_DN_S	SSL_CLIENT_S_DN_D
SSL_CLIENT_I_DN_C	SSL_CLIENT_S_DN_Email	SSL_CLIENT_I_DN
SSL_CLIENT_I_DN_O	SSL_CLIENT_I_DN_ST	SSL_CLIENT_I_DN_L
SSL_CLIENT_I_DN_T	SSL_CLIENT_I_DN_OU	SSL_CLIENT_I_DN_CN
SSL_CLIENT_I_DN_S	SSL_CLIENT_I_DN_I	SSL_CLIENT_I_DN_G
SSL_CLIENT_I_DN_Email	SSL_CLIENT_I_DN_D	SSL_CLIENT_I_DN_UID
SSL_CLIENT_CERT	SSL_CLIENT_CERT_CHAIN_n	SSL_CLIENT_ROOT_CERT
SSL_CLIENT_VERIFY	SSL_CLIENT_M_VERSION	SSL_SERVER_M_VERSION
SSL_SERVER_V_START	SSL_SERVER_V_END	SSL_SERVER_M_SERIAL
SSL_SERVER_S_DN_C	SSL_SERVERT_S_DN_ST	SSL_SERVER_S_DN
SSL_SERVER_S_DN_OU	SSL_SERVER_S_DN_CN	SSL_SERVER_S_DN_O
SSL_SERVER_S_DN_I	SSL_SERVER_S_DN_G	SSL_SERVER_S_DN_T
SSL_SERVER_S_DN_D	SSL_SERVER_S_DN_UID	SSL_SERVER_S_DN_S
SSL_SERVER_I_DN	SSL_SERVER_I_DN_C	SSL_SERVER_S_DN_Email
SSL_SERVER_I_DN_L	SSL_SERVER_I_DN_O	SSL_SERVER_I_DN_ST
SSL_SERVER_I_DN_CN	SSSL_SERVER_I_DN_T	SSL_SERVER_I_DN_OU
SSL_SERVER_I_DN_G	SSL_SERVER_I_DN_I	

SSLRequireSSL

Denies access to clients not using SSL. This is a useful directive for absolute protection of a SSL-enabled virtual host or directories in which configuration errors could create security vulnerabilities.

Category	Value
Syntax	SSLRequireSSL
Default	None
Context	directory

SSLSessionCache

Specifies the global/interprocess session cache storage type. The cache provides an optional way to speed up parallel request processing.

Category	Value
Valid Values	■ `none`: disables the global/interprocess session cache. Produces no impact on functionality, but makes a major difference in performance. ■ `shmht:`/path/to/datafile[bytes]: Uses a high-performance hash table (`bytes` specifies approximate size) inside a shared memory segment in RAM, which is established by the `/path/to/datafile`. This hash table synchronizes the local SSL memory caches of the server processes. ■ `shmcb:`/path/to/datafile[bytes]: Uses a high-performance Shared Memory Cyclic Buffer (SHMCB) session cache to synchronize the local SSL memory caches of the server processes. The performance of `shmcb` is more uniform in all environments when compared to `shmht`.
Syntax	`SSLSessionCache` *`type`*

Examples	`SSLSessionCache shmht: /`*`ORACLE_HOME`*`/Apache/Apache/logs/ssl_scache(512000)`
	`SSLSessionCache shmcb: /`*`ORACLE_HOME`*`/Apache/Apache/logs/ssl_scache(512000)`

Default	`SSLSessionCache` *none*

SSLSessionCacheTimeout

Specifies the number of seconds before a SSL session in the session cache expires.

Category	Value
Syntax	`SSLSessionCacheTimeout` *`seconds`*
Default	300
Context	server configuration

SSLVerifyClient

Specifies whether or not a client must present a certificate when connecting.

Category	Value
Valid Values	■ `none`: No client certificate is required ■ `optional`: Client may present a valid certificate ■ `require`: Client must present a valid certificate
Syntax	`SSLVerifyClient` *`level`*
Default	None
Context	server configuration, virtual host

Note: The level `optional_no_ca` included with `mod_ssl` (in which the client can present a valid certificate, but it need not be verifiable) is not supported in `mod_ossl`.

SSLWallet

Specifies the location of the wallet with its **WRL**.

Category	Value
Syntax	`SSLWallet` *`wrl`* The format of `wrl` is: *`file:path to wallet`*
Example	`SSLWallet` *`file:`*`/etc/ORACLE/WALLETS/`*`server`* Other values of wrl may be used as permitted by the Oracle SSL product.
Default	None
Context	server configuration, virtual host

SSLWalletPassword

Specifies the Wallet password needed to access the wallet specified within the same context. You can choose either a **cleartext** wallet password or an obfuscated password. The obfuscated password is created with the command line tool `iasobf`. If you must use a regular wallet, Oracle recommends that you use the obfuscated password instead of a cleartext password.

See Also: "Using the iasobf Utility" on page 9-15

Category	Value
Syntax	`SSLWalletPassword` *`password`* If no password is required do not set this directive. Note: If a wallet created with the Auto Login feature of Oracle Wallet Manager is used, then do not set this directive because these wallets do not require passwords.
Default	None
Context	server configuration, virtual host

Note: `SSLWalletPassword` has been deprecated. A warning message is generated in the Oracle HTTP Server log if this directive is used. For secure wallets, Oracle recommends that you get a SSO wallet, with auto-login enabled, instead. Refer to the"Task 1: Creating a Real Wallet" on page 9-1.

Using the iasobf Utility

The `iasobf` utility enables you to generate an obfuscated wallet password from a **cleartext** password.

If you are using an Oracle Wallet that has been created with Auto Login enabled (an SSO wallet), then you do not need to use this utility. However, if you must use a regular wallet with a password, then Oracle recommends that you use the password obfuscation tool `iasobf`, which is located in *`ORACLE_HOME`*`/Apache/Apache/bin`, to generate an obfuscated wallet password from a cleartext password.

To generate an obfuscated wallet password, the command syntax is:

```
iasobf -p password
```

The obfuscated password is printed to the terminal. `iasobf` requires operating system user of httpd process. Accordingly, use the `root` argument for UNIX or `system` argument for Windows. For example, on UNIX, the command will be `iasobf -password root`.

> **Note:** The corresponding tool for Windows environments is called `osslpassword`, which can be used in the same way as `iasobf`.

Using mod_proxy Directives

The following directives are for mod_proxy support only:

- SSLProxyCache
- SSLProxyCipherSuite
- SSLProxyProtocol
- SSLProxyWallet
- SSLProxyWalletPassword

SSLProxyCache

Specifies whether the proxy cache will be used. The proxy will use the same session as the SSL server uses.

Category	Value
Syntax	`SSLProxyCache` *`on/off`*
Default	`SSLProxyCache off`
Context	server configuration, virtual host

SSLProxyCipherSuite

Specifies the proxy server's cipher suite.

Category	Value
Syntax	`SSLCipherSuite` *`cipher-spec`*
Default	`None`
Context	server configuration, virtual host

SSLProxyProtocol

Controls the proxy server's SSL protocol flavors.

Category	Value
Syntax	`SSLProxyProtocol [+-]` *`protocol`*
Default	`None`
Context	server configuration, virtual host

SSLProxyWallet

Specifies the location of the wallet containing the certificates to use when opening proxy connections.

Category	Value
Syntax	`SSLProxyWallet` *`wrl`*
Default	`None`
Context	server configuration, virtual host

SSLProxyWalletPassword

Specifies the proxy wallet password.

Category	Value
Syntax	`SSLProxyWalletPassword` *`password`*
Default	`None`
Context	server configuration, virtual host

Note: `SSLProxyWalletPassword` has been deprecated. A warning message is generated in the Oracle HTTP Server log if this directive is used. For secure wallets, Oracle recommends that you get a SSO wallet instead.

A

Load Balancing Using mod_oc4j

This chapter contains information about `mod_oc4j` load balancing, including metric-based load balancing. Topics include:

- Load Balancing Policies
- Load Balancing Parameters
- Metric-based Load Balancing

Load Balancing Policies

This section contains information about load balancing policies that `mod_oc4j` supports:

- Random
- Round Robin
- Random with Local Affinity
- Round Robin with Local Affinity
- Random using Routing Weight
- Round Robin using Routing Weight
- Metric Based
- Metric Based with Local Affinity

Random

`mod_oc4j` randomly selects an OC4J instance from a list of OC4J instances that are candidates to service a request.

Round Robin

`mod_oc4j` randomly selects an OC4J instance from an ordered list of OC4J instances that are candidates to service a request. Other OC4J instances are selected from the ordered list in turn, until the initially selected server is selected again. This sequence is repeated. If a particular OC4J instance is stopped or is unavailable, then that instance is skipped (no attempt is made to select it) until it can be brought back in service.

Random with Local Affinity

mod_oc4j randomly selects local OC4J processes to service requests. When no local OC4J processes are available, mod_oc4j randomly selects remote OC4J processes and gives them equal opportunity to be selected.

Round Robin with Local Affinity

mod_oc4j routes all requests to local OC4J processes in a round robin manner. When no local processes are available, mod_oc4j routes requests equally to each OC4J process on different hosts.

Random using Routing Weight

mod_oc4j distributes requests according to the routing weight configured for each host. One OC4J process is selected randomly from the OC4J processes on that host.

Round Robin using Routing Weight

mod_oc4j distributes the total request load to OC4J processes on each host based on the routing weight configured to each host. mod_oc4j selects an OC4J process in round robin manner from the OC4J processes on that host.

Metric Based

mod_oc4j routes requests based on run time metrics from OC4J processes that indicate how much load can be placed on the OC4J process.

Metric Based with Local Affinity

mod_oc4j routes all requests to local OC4J processes based on the run time performance metrics of OC4J processes. When there are no local OC4J processes available, mod_oc4j routes requests to each OC4J process on different hosts as per their performance metrics only.

Load Balancing Parameters

This section discusses the following load balancing parameters:

- Oc4jSelectMethod
- Oc4jRoutingWeight

Oc4jSelectMethod

Selects an OC4J instance for load balancing.

Category	Value
Syntax	`Oc4jSelectMethod roundrobin | roundrobin:local | roundrobin:weighted | random | random:local | random:weighted | metric | metric:local`
Required	No
Default	If `Oc4jSelectMethod` is not specified, it defaults to "`Oc4jSelectMethod roundrobin`".

Category	Value

Example	`Oc4jSelecctMethod random:local`
	`Oc4jSelecctMethod metric`

Usage	■ `Oc4jSelectMethod random`: Selects an OC4J process according to "Random" load balancing policy. ■ `Oc4jSelectMethod roundrobin:weighted`: Selects an OC4J process according to "Round Robin using Routing Weight" load balancing policy. ■ `Oc4jSelecctMethod metric:local:` Selects an OC4J process according to "Metric Based with Local Affinity" load balancing policy.

This directive is only applicable to the base server for Oracle Application Server 10*g* Release 2 (10.1.2) and an error will be printed at startup if specified within a `VirtualHost` container.

Oc4jRoutingWeight

Associates a request routing weight for each machine during load balancing. Weighted routing is a load balancing strategy that distributes requests according to a predefined value assigned to each machine based on the predicted ability to handle load.

Category	Value
Syntax	`Oc4jRoutingWeight` <*node_name*> <*routing_weight*>
Required	No
Default	It defaults to OC4J processes on all the nodes with routing weight as 1. If `Oc4jRoutingWeight` is specified, but some hosts are not specified, it defaults to OC4J processes on any non-specified node with routing weight as 1.

Category	Value

Example	■ There are two hosts in an Oracle Application Server cluster: Host_A and Host_B. Each has Oracle HTTP Server and OC4J processes running on them.
	`Oc4jSelectMethod random:local`
	`Oc4jRoutingWeight Host_A 3`
	`Oc4jRoutingWeight Host_B 2`
	`Oc4jRoutingWeight` directives are ignored. `mod_oc4j` on Host_A randomly routes all requests to OC4J processes on Host_A, `mod_oc4j` on Host_B randomly routes all requests to OC4J processes on Host_B.
	■ There are four hosts in an Oracle Application Server cluster: Host_A, Host_B, Host_C, and Host_D. Each has Oracle HTTP Server and OC4J processes running on them.
	`Oc4jSelectMethod roundrobin:weighted`
	`Oc4jRoutingWeight Host_A 3`
	`Oc4jRoutingWeight Host_B 2`
	`mod_oc4j` on all the machines route three times the number of requests to OC4J processes running on Host_A, two times the number of requests on Host_B, one time the number of requests on Host_C, and one time the number of requests on Host_D in a round robin manner.
	■ There are four hosts in an Oracle Application Server cluster: Host_A, Host_B, Host_C, and Host_D. Each has Oracle HTTP Server and OC4J processes running on them.
	`Oc4jSelectMethod roundrobin:weighted`
	`mod_oc4j` on all the machines route requests equally to OC4J processes on Host_A, Host_B, Host_C, and Host_D in a round robin manner.

Usage	`Oc4jRoutingWeight` is taken into account only when Oc4jSelectMethod specifies weighted. "`Oc4jRoutingWeight` *<node_name> <routing_weight>*" associates a request routing weight to each node. `node_name` can be in host name or IP address format. For hosts with multiple interfaces, if different interfaces are specified, it is assumed that they are different hosts.

This directive is only applicable to the base server for Oracle Application Server 10*g* Release 2 (10.1.2) and an error will be printed at startup if specified within a `VirtualHost` container.

Metric-based Load Balancing

Metric-based load balancing is a way to distribute request load among OC4Js based on a "health" metric that each OC4J reports. The metric range is between 0 and 100, where 0 is very busy, or unhealthy, and 100 is not busy, or healthy. When metric-based load balancing is enabled, requests are distributed among OC4Js based on a ratio of a metric received for an individual OC4J, divided by the total of the metrics received from all the OC4Js.

For example, OC4J process p1 reports a metric of 20, process p2 reports a metric of 40, and process p3 reports a metric of 90. The requests would be distributed as follows:

- p1 is routed 20 out of every 150 requests (13%)
- p2 is routed 40 out of every 150 requests (27%)
- p3 is routed 90 out of every 150 requests (60%)

You must configure Oracle HTTP Server and OC4J to enable metric-based load balancing. The following sections contain the required configuration information:

- Configuring Oracle HTTP Server
- Configuring OC4J

Configuring Oracle HTTP Server

On the Oracle HTTP Server side, specify "`Oc4jSelectMethod metric`" or "`Oc4jSelectMethod metric:local`" in mod_oc4j.conf.

See Also:

- "Oc4jSelectMethod" on page A-2
- "Metric Based" on page A-2
- "Metric Based with Local Affinity" on page A-2

Configuring OC4J

On the OC4J side, you must configure the metric collector in *`ORACLE_HOME`*`/j2ee/home/config/server.xml` in UNIX or *`ORACLE_HOME`*`/j2ee\home\config\server.xml` on Windows. Configuring the `<metric-collector>` element tells OC4J to start sending a metric to `mod_oc4j` so that `mod_oc4j` can make routing decisions to load balance incoming requests to a list of available OC4J instances.

The metric sent from OC4J to `mod_oc4j` is used only when metric-based load balancing is specified for `mod_oc4j` and when OC4J runs in an Oracle Application Server environment.

If you specify metric-based load balancing in `mod_oc4j` and do not specify the `<metric-collector>` element in `server.xml`, then `mod_oc4j` expects OC4J to send metrics, but OC4J does not send metrics. In this case, `mod_oc4j` reports the following warning message:

```
No run time metrics for oc4j(opmnid=%s) in notification Oc4jSelectMethod is
configured to use run time metrics, please make sure OC4J side is configured
accordingly. Default to 50.
```

In this case, `mod_oc4j` uses the value "50" for each of the OC4J processes and continues.

Likewise, if you specify the `<metric-collector>` element in `server.xml`, but do not specify metric-based load balancing in `mod_oc4j`, then OC4J sends metrics but `mod_oc4j` is not configured to receive metrics. In this case, `mod_oc4j` ignores the metrics and uses whatever the configured method is for load balancing. You specify the load balancing method with Oc4jSelectMethod. If no `Oc4jSelectMethod` is specified, then `mod_oc4j` uses the default, which is roundrobin.

All OC4Js that are used from this Oracle HTTP Server instance must be configured identically. Otherwise, the number returned from the OC4Js will not be comparable and can produce some very poor load balancing results.

When `mod_oc4j` receives a notification containing the metrics information from OC4J, it immediately changes the request routing behavior. The default interval between notifications from OC4J is 30 seconds. This value can be configured using the system property `opmnPingInterval`, which is passed on the command line when OC4J is started by OPMN. To change the interval between notifications, specify the following in `opmn.xml` under the `OC4J <process-set>` configuration element:

```
<module-data>
  <category id="start-parameters">
    <data id="java-options" value="-DopmnPingInterval=<new ping interval value>"/>
  </category>
</module-data>
```

See Also: *Oracle Process Manager and Notification Server Administrator's Guide*

Specifying Metrics for OC4J

The following two methods can be used to specify metrics for OC4J:

- Configuring Metric-based Load Balancing to Use the DMSMetricCollector
- Building Your Own Metric Collector

Configuring Metric-based Load Balancing to Use the DMSMetricCollector

In this out of the box method, the `<metric-collector>` element takes a single attribute: `classname`. This attribute defines an interface for gathering and calculating a server-wide metric. Use `oracle.oc4j.server.DMSMetricCollector` for the `classname` attribute when using a DMS-based metric collector. A `DMSMetricCollector` instance takes several parameters.

The DMS metric is specified using the 'dms-noun' parameter, which is shown in the configuration example below. This is the metric on which the `DMSMetricCollector` bases its calculation. The recommended DMS metric for metric-based load balancing is `/oc4j/default/WEBS/processRequest.time`. This metric represents the processing time of the servlets in the default Web application.

The value sent to OC4J is a weighted average of the value computed based on the current DMS metric value, and the last value computed the last time a value was sent. The default weight is 0.7 for the current value, and 0.3 for the previous value. To modify the weights, one may set the "`history-proportion`" as shown the following example. This results in a weight of 0.8 for the current value and 0.2 for the previous value.

```
<metric-collector classname="oracle.oc4j.server.DMSMetricCollector">
   <init-param>
     <param-name>
       dms-noun
     </param-name>
     <param-value>
       /oc4j/default/WEBs/processRequest.time
     </param-value>
   </init-param>
   <init-param>
     <param-name>
       history-proportion
     </param-name>
     <param-value>
       0.2
     </param-value>
   </init-param>
   <init-param>
     <param-name>
       debug
     </param-name>
     <param-value>
```

```
       false
     </param-value>
   </init-param>
</metric-collector>
```

> **See Also:** *Oracle Application Server Performance Guide* for a list of DMS metrics.

How DMS Metrics are Converted for Metric-based Load Balancing When `getMetrics()` is called, the value of the DMS metric specified by the `dms-noun` parameter is obtained. The `delta` with the previous measurement is computed.

Since the scale is from 0 to 100, and the measurement is potentially unbounded, the following formula is applied:

```
metric = 100 / (1 + (log (1 + delta)))
```

As outlined above, this metric should match the current situation, but also reflect on previous metric history. You can assign a weight of 1/3 to the previous history and 2/3 for the average just collected in order to form the new metric.

This new metric becomes the next metric's history, and factors in less and less as time passes. As shown in the configuration example, you can specify the proportion of the previous metric that enters in new metrics, by setting the `history-proportion` parameter to a floating point value between 0.0 and 1.0. Higher values mean that historical values matter more, making the metric less volatile. A lower value makes the metric reflect the most recent metric. The metric returned is as follows:

```
smoothedmetric = ((1 - history-proportion) * metric) + (history-proportion *
previousmetric)
```

The `smoothedmetric` is what is sent to `mod_oc4j` for load balancing purposes.

Building Your Own Metric Collector

You can implement the interface oracle.oc4j.api.MetricCollector to supply your own metric collector to `mod_oc4j`. The metric has to be between 0 and 100.

All metric collectors must implement the interface `oracle.oc4j.api.MetricCollector`. The concrete metric collector must have a constructor with no parameters so it can be instantiated.

The schema elements for the feature are in `server.xml`, and look like:

```
<metric-collector classname="my.package.name.MyClassName">
  <init-param>
    <param-name>
      mysetting
    </param-name>
    <param-value>
      12345
    </param-value>
  </init-param>
</metric-collector>
```

As per the preceding example, 0 or more parameters can be set on the metric collector, and are passed to it when `setParameters()` method is called. `setEnabled(true)` is called once after `setParameters()`. It signals the metric collector that it can begin gathering data as needed.

> **Note:** If the custom metric collector starts threads, the threads need to be daemon threads. Otherwise they may prevent the server from shutting down in an orderly fashion.

After `oracle.oc4j.api.MetricCollector` has been implemented, package your metric in a jar file and add it to your library path, using the following, in `server.xml`:

```
<library path="<path to>/mymetric.jar"/>
```

oracle.oc4j.api.MetricCollector Here is the `oracle.oc4j.api.MetricCollector` interface:

```
package oracle.oc4j.api;

 import java.util.Map;

 /**
  * Defines an interface for gathering and obtaining a server-wide metric.
  * The metric is used in iAS mode, by mod_oc4j, to load balance between
  * virtual oc4j instances.
  * The metric value is relative, and should be between 0 and 100, both
  * inclusive.
  * When configured for metric load balancing,
  * Mod_oc4j will route preferably to an oc4j with the greater value.
  * <p>Concrete instances of this class must have a public empty constructor in
  * order to be loaded and instantiated.
  */
 public interface MetricCollector {
   /**
    * Support for debugging: This is a property name to set to true in order
    * to display the metric that is sent from the server
    */
   String OC4J_METRIC_DEBUG_PROPERTY = "oc4j.metric.debug";

   /**
    * Debugging flag that depends on @{link #OC4J_METRIC_DEBUG_PROPERTY}
    */
   boolean DEBUG = Boolean.getBoolean( OC4J_METRIC_DEBUG_PROPERTY );

   /**
    * Initial metric to return when no measurement has been made (property key)
    */
   String OC4J_INITIAL_METRIC_PROPERTY = "oc4j.metric.initial";
   /**
    * Initial metric to return when no measurement has been made.
    * Default is 50
    */
   int INITIAL_METRIC = Integer.getInteger( OC4J_INITIAL_METRIC_PROPERTY, 50
).intValue();

   /**
    * Enabled flag for the collector.
    * @return true if the collector is collecting data
    */
   boolean isEnabled();

   /**
   */
  void setEnabled( boolean enabled);

  /**
   * The parameters the metric collector is configured with.
   * This method will be called even when the set of parameters is null.
   * @param params the key/value pairs the metric collector is configured with,
   * or <code>null</code> if none
   */
  void setParameters( Map params );

  /**
   * @return a metric between 0 and 100, inclusive. 100 is better, 0 is worse
   */
  int getMetric();

}
```

B

Configuration Files

This appendix lists commonly used Oracle HTTP Server configuration files.

Files discussed are:

- dms.conf
- httpd.conf
- iaspt.conf
- mime.types
- mod_oc4j.conf
- mod_osso.conf
- opmn.xml
- oracle_apache.conf
- php.ini
- ssl.conf

dms.conf

Enables you to monitor performance of site components with Oracle's Dynamic Monitoring Service (DMS).

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf`
- Windows: `ORACLE_HOME\ohs\conf`

See Also: *Oracle Application Server Performance Guide*

httpd.conf

This is a server configuration file which typically contains directives that affect how the server runs, such as user and group IDs it should use, and location of other files. Because the server configuration file is the main file that the server starts with, Oracle HTTP Server does not include any directive that says where to locate it. The location is passed on command line when the server starts.

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf/httpd.conf`
- Windows: `ORACLE_HOME\ohs\conf\httpd.conf`

You should use only this file, and not srm.conf or access.conf because it is much easier to manage a single configuration file.

Note: If you have an Oracle Application Server installation in /home/your_directory/orahome and it is linked to /private/your_directory/orahome, the files in the installation are accessible from either /home/your_directory/orahome or /private/your_directory/orahome.

After installation, the httpd.conf file contains an entry for the dms.conf file that uses the original Oracle home path. For example:

```
include /home/your_directory/orahome/Apache/Apache/conf/dms.conf
```

Do not replace the original Oracle home path with the linked Oracle home path.

httpd.conf File Structure

httpd.conf is arranged in the following sections:

- Global Environment
- Main Server Configuration
- Virtual Hosts Parameters

Global Environment

This is section one of the httpd.conf file. It contains configuration directives dealing with Oracle HTTP Server.

See Also:

- "Specifying File Locations" on page 3-3
- "Configuring the Number of Processes and Connections" on page 4-2
- "Specifying Listener Ports and Addresses" on page 5-1

Main Server Configuration

This is section two of the httpd.conf file. It contains the directives of the default server.

See Also: "Setting Server and Administrator Functions" on page 3-1.

Virtual Hosts Parameters

This is section three of the httpd.conf file. It contains parameters specific to virtual hosts, which override some of the main server configuration defaults.

iaspt.conf

Configures the port tunneling process. Port tunneling allows all communication between Oracle HTTP Server and OC4J to happen on a single, or a small number of ports.

It is located at:

- UNIX: `ORACLE_HOME/iaspt/conf`
- Windows: `ORACLE_HOME\iaspt\conf`

> **See Also:** "Understanding Port Tunneling" on page 8-7

mime.types

Controls the Multi Internet media types that are sent to the client for the given file extensions. Sending the correct media type to the client is important so that the client knows how to handle the content of the file. You can add extra types in the mime type file or add an `AddType` directive in the configuration file.

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf`
- Windows: `ORACLE_HOME\ohs\conf`

> **See Also:** "mod_mime" on page 7-8

mod_oc4j.conf

Configures and loads the `mod_oc4j` module, and is enabled by default. It routes requests from Oracle HTTP Server to OC4J, and therefore contains routing information.

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf`
- Windows: `ORACLE_HOME\ohs\conf`

> **See Also:** "mod_oc4j" on page 7-8

mod_osso.conf

Configures `mod_osso`, which enables single sign-on for Oracle HTTP Server.

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf`
- Windows: `ORACLE_HOME\ohs\conf`

> **See Also:** "mod_osso" on page 7-18

opmn.xml

Describes the processes that Oracle Process Manager and Notification Server (OPMN) manages within an Oracle Application Server installation.

The `opmn.xml` file is the main configuration file for OPMN. It contains information for the ONS, the PM, and Oracle Application Server component-specific configuration.`opmn.xml` shows you which Oracle Application Server components OPMN is managing on your system. It contains Oracle Application Server component entries arranged in the following hierarchical structure:

```
<ias-component>
  <process-type>
    <process-set>
```

- **<ias-component>**: This entry represents the Oracle Application Server component. It enables management of the component for processes such as starting and stopping.
- **<process-type>**: This subcomponent of the `<ias-component>` entry declares the type of process to run by association with a specific PM module.
- **<process-set>**: This sub-subcomponent of the `<ias-component>` entry enables you to declare different sets of optional runtime arguments and environments for the Oracle Application Server component.

`opmn.xml` is located at:

- UNIX: `ORACLE_HOME/opmn/conf`
- Windows: `ORACLE_HOME\opmn\conf`

> **See Also:** *Oracle Process Manager and Notification Server Administrator's Guide*

oracle_apache.conf

Stores configuration files of supported modules.

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf`
- Windows: `ORACLE_HOME\ohs\conf`

> **Note:** For the Oracle Application Server Infrastructure install type, another configuration file is included by `oracle_apache.conf` called `oracle_ocm.conf`. It contains configuration for Oracle Application Server Certificate Authority.

php.ini

Configures `mod_php`. This file should not be renamed as PHP looks for this specific file name.

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf`
- Windows: `ORACLE_HOME\ohs\conf`

> **See Also:** "mod_php" on page 7-20

ssl.conf

Includes the SSL definitions and virtual host container. Out of the box, it is disabled by default.

It is located at:

- UNIX: `ORACLE_HOME/ohs/conf`
- Windows: `ORACLE_HOME\ohs\conf`

C

Frequently Asked Questions

This chapter provides answers to frequently asked questions about Oracle HTTP Server.

See Also: "Frequently Asked Questions" in the Apache Server documentation.

Documentation from the Apache Software Foundation is referenced when applicable.

Note: Readers using this guide in PDF or hard copy formats will be unable to access third-party documentation, which Oracle provides in HTML format only. To access the third-party documentation referenced in this guide, use the HTML version of this guide and click the hyperlinks.

Creating Application-specific Error Pages

Oracle HTTP Server has a default content handler for dealing with errors. You can use the `ErrorDocument` directive to override the defaults.

See Also: ErrorDocument directive in the Apache Server documentation.

Offering HTTPS to ISP (Virtual Host) Customers

For HTTP, Oracle HTTP Server supports two types of virtual hosts: name-based and IP-based. HTTPS supports only IP-based virtual hosts.

If you are using IP-based virtual hosts for HTTP, then the customer has a virtual server listening on port 80 of a per-customer IP address. To provide HTTPS for these customers, simply add an additional virtual host per user listening on port 4443 of that same per-customer IP address and use SSL directives, such as Using mod_ossl Directives to specify the per-customer SSL characteristics. Note that each customer can have their own wallet and server certificate.

If you are using name-based virtual hosts for HTTP, each customer has a virtual server listening on port 80 of a shared IP address. To provide HTTPS for those customers, you can add a single shared IP virtual host listening on port 4443 of the shared IP address. All customers will share the SSL configuration, including the wallet and ISP's server certificate.

See Also: "Running Oracle HTTP Server as Root" on page 4-4

Using Different Language and Character Set Versions of Document

You can use *multiviews*, a general name given to the Apache server's ability to provide language and character-specific document variants in response to a request.

Sending Proxy Sensitive Requests to HTTP Server Behind a Firewall

You should use the Proxy directives, and not the Cache directives, to send proxy sensitive requests across firewalls.

mod_oc4j Information

mod_oc4j is a module that integrates with Web servers, typically Oracle HTTP Server, and routes request to the backend OC4J processes. OPMN module keeps mod_oc4j aware of the status of different OC4J processes, so mod_oc4j routes only to the processes that are up and running. mod_oc4j also understands the concepts of Oracle Application Server Clusters and OC4J islands, and routes accordingly to provide as much transparent failover as possible.

See Also: "mod_oc4j" on page 7-8

mod_oc4j Communication to OC4J using SSL

The AJP communication between mod_oc4j and OC4J processes can now be over AJP/SSL. Previously, this was in the clear. Also, the SSL negotiation does not happen each time mod_oc4j and OC4J communicate, resulting in less performance impact.

See Also: "Enabling SSL between mod_oc4j and OC4J" on page 7-15

Oracle HTTP Server Version Number

Oracle HTTP Server is based on Apache version 2.0.52.

Applying Apache Security patches to Oracle HTTP Server

You cannot apply the Apache security patches to Oracle HTTP Server for the following reasons:

- Oracle tests and appropriately modifies security patches before releasing them to Oracle HTTP Server users.
- In many cases those alerts may not be applicable, for example, openSSL alerts, since Oracle has removed those components from the stack in use.
- Oracle releases these patches soon enough that the time-delay impact of getting the patch from Oracle versus open source organization should be minimal and the benefit with respect to supportability, tremendous.

Compressing Output from Oracle HTTP Server

In general, Oracle recommends the use of OracleAS Web Cache for this purpose. There are other freeware modules, such as mod_gzip that may be plugged in for this purpose, but their use is not supported. When using these, there may be an error message with respect to EAPI, but in general that can be ignored.

Supporting PHP

mod_php is fully supported in Release 2 (10.1.2.)

See Also: "mod_php" on page 7-20

Protecting Web Site From Hackers

There are many attacks, and new attacks are invented everyday. The following are some general guidelines for securing your site. You can never be completely secure, but you can avoid being an easy target.

- Use a commercial firewall, such as Checkpoint FW-1 or Cisco PIX between your ISP and your Web server. Recognize, however, that not all hackers are outside your organization.
- Use switched ethernet to limit the amount of traffic a compromised server can sniff. Use additional firewalls between Web server machines and highly sensitive internal servers running database and enterprise applications.
- Remove unnecessary network services such as RPC, Finger, telnet from your server machine.
- Carefully validate all input from Web forms. Be especially wary of long input strings and input that contains non-printable characters, HTML tags, or javascript tags.
- Encrypt or randomize the contents of cookies that contain sensitive information. For example, it should be difficult to guess a valid sessionID to prevent a hacker from hijacking a valid session.
- Check often for security patches for all your system and application software, and install them as soon as possible. Be sure these patches come from bona fide sources; download from trusted sites and verify the cryptographic checksum.
- Use an intrusion detection package to monitor for defaced Web pages, viruses, and presence of "rootkits" that indicate hackers have broken in. If possible, mount system executables and Web content on read-only file systems.
- Have a "forensic analysis" package on hand to capture evidence of a break in as soon as detected. This aids in prosecution of the hackers.

D

Troubleshooting Oracle HTTP Server

This appendix describes common problems that you might encounter when using Oracle HTTP Server, and explains how to solve them.

It contains the following topics:

- Problems and Solutions
- Need More Help?

Problems and Solutions

This section describes common problems and solutions. It contains the following topics:

- Intermittent HTTP-500 errors
- Firewall Between Oracle HTTP Server and OC4J Blocks Connections
- Oracle HTTP Server Unable to Start Due to Port Conflict
- Machine Overloaded by Number of HTTPD Processes
- Permission Denied When Starting Oracle HTTP Server on Port Below 1024
- Oracle HTTP Server May Fail To Start If PM Files Are Not Located Correctly
- SSO Client Authentication Fails with Webcache Reverse Proxy

Intermittent HTTP-500 errors

Certain Microsoft Internet Explorer security patches have resulted in intermittent HTTP-500 errors, such as `MOD_OC4J_0145`, `MOD_OC4J_0119`, `MOD_OC4J_0013` errors, when the `KeepAlive` directive is set on "On" in Oracle HTTP Server.

Problem

Intermittent HTTP-500 errors caused by bug in Microsoft Internet Explorer.

Solution

There are two possible solutions for this problem:

- Patch all the client Internet Explorer browsers.
- If the previous option is not practical, set `KeepAlive` to "Off" in Oracle HTTP Server.

Consult Metalink Note 269980.1 on `http://metalink.oracle.com` for details regarding this issue. The easiest way to access the note is to click the Advanced Search button at the top of the Oracle*Metalink* site, and search for Doc ID "269980.1".

See Also: "KeepAlive" on page 5-3

Firewall Between Oracle HTTP Server and OC4J Blocks Connections

Oracle HTTP Server is unable to forward requests to OC4J when certain firewalls are used between them.

Problem

Oracle HTTP Server processes maintain persistent connections with OC4J processes. If the firewall times out a connection before Oracle HTTP Server does, then requests to the OC4J processes can result in errors, or can take a very long time, depending on how the firewall and the operating system are configured.

Solution

Set the Oracle HTTP Server directive `OC4JConnTimeout` to a value less than that of the firewall timeout (this is firewall specific).

See Also: "Oc4jConnTimeout" on page 7-10

Oracle HTTP Server Unable to Start Due to Port Conflict

You can get the following error if Oracle HTTP Server is unable to start due to port conflict:

```
[crit] (98) Address already in use: make_sock: could not bind to
port 7778
```

Problem

Oracle HTTP Server is unable to start as its port number is being used by another process.

Solution

Determine what process is already using the port by pointing a browser at the address assigned to Oracle HTTP Server and viewing the results. Depending on the results, either change the IP:port address of Oracle HTTP Server, or that of the conflicting process.

Machine Overloaded by Number of HTTPD Processes

When there are too many httpd processes running on a machine, the response time plummets.

Problem

When too many httpd processes are started, there are insufficient resources for normal processing.

Solution

Lower value of `MaxClients` to a value the hardware box can accommodate.

See Also: "MaxClients" on page 4-3

Permission Denied When Starting Oracle HTTP Server on Port Below 1024

You will get the following errors if you try to start Oracle HTTP Server on port below 1024:

Bind errors on ports below 1024: `PERMISSION DENIED: MAKE_SOCK: COULD NOT BIND TO PORT 443`.

Problem

Oracle HTTP Server will not start on ports below 1024 because `root` privileges are needed to bind these ports. Also, steps to configure `.apachectl` have not been followed.

Solution

Perform the following steps to enable Oracle HTTP Server to run as root on ports below 1024:

1. Log in as `root`.
2. Run the following commands in the middle-tier Oracle home:

```
cd ORACLE_HOME/Apache/Apache/bin
chown root .apachectl
chmod 6750 .apachectl
```

Oracle HTTP Server May Fail To Start If PM Files Are Not Located Correctly

Oracle HTTP Server may encounter the following error, and fail to start:

```
"[error] Can't locate mod_perl.pm in @INC (@INC contains:$ORACLE_HOME/perl/...)
```

or,

```
[error] Can't locate Apache::Registry.pm in @INC (@INC contains: $ORACLE_
HOME/per/...)
```

Problem

`mod_perl` needs to locate PM files kept under the *ORACLE_HOME*`/ohs/mod_perl` directory. Without these PM files, `mod_perl` will not start.

Solution

For UNIX, check that `apachectl` has correctly defined the variable `PERL5LIB`. It should point to the following:

```
ORACLE_HOME/perl/lib/5.8.3:ORACLE_HOME/perl/lib/site_perl/5.8.3:ORACLE_
HOME/ohs/mod_perl/lib/site_perl/5.8.3/sun4-solaris-thread-multi
```

For Windows, check that the environment sub-section in the HTTP Server section in `opmn.xml` has a correct entry for `PERL5LIB`. It should point to the following:

```
$ORACLE_HOME\ohs\mod_perl€ib\MSWin32-x86-multi-thread;$ORACLE_
HOMEerl\5.8.3€ib;$PERL5LIB
```

SSO Client Authentication Fails with Webcache Reverse Proxy

SSO client authentication fails with Webcache reverse proxy.

Problem

During SSO client login, the client certificate should be authenticated from the browser with the SSO server and connect successfully. However, it fails because the `ssoServer.log` shows it is trying to authenticate the certificate stored in the Webcache wallet and not the one from the browser.

Solution

Perform the following steps:

1. Edit `$ORACLE_HOME/Apache/Apache/conf/httpd.conf` and make sure it has the following:

   ```
   LoadModule certheaders_module libexec/mod_certheaders.so
   AddCertHeader HTTPS
   AddCertHeader SSL_CLIENT_CERT
   ```

2. Edit the `$ORACLE_HOME/sso/conf/sso_apache.conf`, and comment out the following line:

   ```
   #SSLOptions +ExportCertData +StdEnvVars
   ```

3. Run `dcmctl updateconfig -ct ohs`
4. Run `opmnctl restartproc type=ohs`
5. Test that the SSO server can be logged into with client authentication.

Need More Help?

You can find more solutions on Oracle*MetaLink*, `http://metalink.oracle.com`. If you do not find a solution for your problem, log a service request.

> **See Also:** *Oracle Application Server Release Notes*, available on the Oracle Technology Network: `http://www.oracle.com/technology/documentation/index.html`

E

Third Party Licenses

This appendix includes the Third Party License for all the third party products included with Oracle Application Server.

Topics discussed are:

- Apache HTTP Server
- Apache SOAP
- DBI Module
- Perl
- PHP
- FastCGI

Apache HTTP Server

Under the terms of the Apache license, Oracle is required to provide the following notices. However, the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Apache software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Apache software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Apache.

The Apache Software License

```
/* ====================================================================
                                 Apache License
                           Version 2.0, January 2004
                        http://www.apache.org/licenses/

   TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

   1. Definitions.

      "License" shall mean the terms and conditions for use, reproduction,
      and distribution as defined by Sections 1 through 9 of this document.

      "Licensor" shall mean the copyright owner or entity authorized by
      the copyright owner that is granting the License.

      "Legal Entity" shall mean the union of the acting entity and all
      other entities that control, are controlled by, or are under common
```

control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made,

use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and
   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

   You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

Apache SOAP

Under the terms of the Apache license, Oracle is required to provide the following notices. However, the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Apache software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Apache software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Apache.

Apache SOAP License

Apache SOAP license 2.3.1

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License, Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of,

publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

   (a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License. You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License.

   You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

DBI Module

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from DBI. Under the terms of the DBI license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the DBI software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the DBI software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or DBI.

The DBI module is Copyright (c) 1994-2002 Tim Bunce. Ireland. All rights reserved.

You may distribute under the terms of either the GNU General Public License or the Artistic License, as specified in the Perl README file.

Perl Artistic License

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions

"Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.
2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.
3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:
   a. place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.
   b. use the modified Package only within your corporation or organization.

   c. rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.

   d. make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

   a. distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.

   b. accompany the distribution with the machine-readable source of the Package with your modifications.

   c. give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.

   d. make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.

6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package through the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.

7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.

8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.

9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.

10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED

WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

Perl

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from Perl. Under the terms of the Perl license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the Perl software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the Perl software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or Perl.

Perl Kit Readme

Copyright 1989-2001, Larry Wall

All rights reserved.

This program is free software; you can redistribute it and/or modify it under the terms of either:

1. the GNU General Public License as published by the Free Software Foundation; either version 1, or (at your option) any later version, or
2. the "Artistic License" which comes with this Kit.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See either the GNU General Public License or the Artistic License for more details.

You should have received a copy of the Artistic License with this Kit, in the file named "Artistic". If not, I'll be glad to provide one.

You should also have received a copy of the GNU General Public License along with this program in the file named "Copying". If not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307, USA or visit their Web page on the internet at http://www.gnu.org/copyleft/gpl.html.

For those of you that choose to use the GNU General Public License, my interpretation of the GNU General Public License is that no Perl script falls under the terms of the GPL unless you explicitly put said script under the terms of the GPL yourself. Furthermore, any object code linked with perl does not automatically fall under the terms of the GPL, provided such object code only adds definitions of subroutines and variables, and does not otherwise impair the resulting interpreter from executing any standard Perl script. I consider linking in C subroutines in this manner to be the moral equivalent of defining subroutines in the Perl language itself. You may sell such an object file as proprietary provided that you provide or offer to provide the Perl source, as specified by the GNU General Public License. (This is merely an alternate way of specifying input to the program.) You may also sell a binary produced by the dumping of a running Perl script that belongs to you, provided that you provide or offer to provide the Perl source as specified by the GPL. (The fact that a Perl interpreter and your code are in the same binary file is, in this case, a form of mere aggregation.) This is my interpretation of the GPL. If you still have concerns or difficulties understanding my intent, feel free to contact me. Of course, the Artistic License spells all this out for your protection, so you may prefer to use that.

mod_perl License

```
/* ====================================================================
 * The Apache Software License, Version 1.1
 *
 * Copyright (c) 1996-2000 The Apache Software Foundation.  All rights
 * reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. The end-user documentation included with the redistribution,
 *    if any, must include the following acknowledgment:
 *       "This product includes software developed by the
 *        Apache Software Foundation (http://www.apache.org/)."
 *    Alternately, this acknowledgment may appear in the software itself,
 *    if and wherever such third-party acknowledgments normally appear.
 *
 * 4. The names "Apache" and "Apache Software Foundation" must
 *    not be used to endorse or promote products derived from this
 *    software without prior written permission. For written
 *    permission, please contact apache@apache.org.
 *
 * 5. Products derived from this software may not be called "Apache",
 *    nor may "Apache" appear in their name, without prior written
 *    permission of the Apache Software Foundation.
 *
 * THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED
 * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
 * DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
 * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 * SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 * LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
 * USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
 * ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
 * OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
 * OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 * ====================================================================
 */
```

Perl Artistic License

The "Artistic License"

Preamble

The intent of this document is to state the conditions under which a Package may be copied, such that the Copyright Holder maintains some semblance of artistic control over the development of the package, while giving the users of the package the right to use and distribute the Package in a more-or-less customary fashion, plus the right to make reasonable modifications.

Definitions

"Package" refers to the collection of files distributed by the Copyright Holder, and derivatives of that collection of files created through textual modification.

"Standard Version" refers to such a Package if it has not been modified, or has been modified in accordance with the wishes of the Copyright Holder as specified below.

"Copyright Holder" is whoever is named in the copyright or copyrights for the package.

"You" is you, if you're thinking about copying or distributing this Package.

"Reasonable copying fee" is whatever you can justify on the basis of media cost, duplication charges, time of people involved, and so on. (You will not be required to justify it to the Copyright Holder, but only to the computing community at large as a market that must bear the fee.)

"Freely Available" means that no fee is charged for the item itself, though there may be fees involved in handling the item. It also means that recipients of the item may redistribute it under the same conditions they received it.

1. You may make and give away verbatim copies of the source form of the Standard Version of this Package without restriction, provided that you duplicate all of the original copyright notices and associated disclaimers.

2. You may apply bug fixes, portability fixes and other modifications derived from the Public Domain or from the Copyright Holder. A Package modified in such a way shall still be considered the Standard Version.

3. You may otherwise modify your copy of this Package in any way, provided that you insert a prominent notice in each changed file stating how and when you changed that file, and provided that you do at least ONE of the following:

   a. place your modifications in the Public Domain or otherwise make them Freely Available, such as by posting said modifications to Usenet or an equivalent medium, or placing the modifications on a major archive site such as uunet.uu.net, or by allowing the Copyright Holder to include your modifications in the Standard Version of the Package.

   b. use the modified Package only within your corporation or organization.

   c. rename any non-standard executables so the names do not conflict with standard executables, which must also be provided, and provide a separate manual page for each non-standard executable that clearly documents how it differs from the Standard Version.

   d. make other distribution arrangements with the Copyright Holder.

4. You may distribute the programs of this Package in object code or executable form, provided that you do at least ONE of the following:

   a. distribute a Standard Version of the executables and library files, together with instructions (in the manual page or equivalent) on where to get the Standard Version.

- **b.** accompany the distribution with the machine-readable source of the Package with your modifications.
- **c.** give non-standard executables non-standard names, and clearly document the differences in manual pages (or equivalent), together with instructions on where to get the Standard Version.
- **d.** make other distribution arrangements with the Copyright Holder.

5. You may charge a reasonable copying fee for any distribution of this Package. You may charge any fee you choose for support of this Package. You may not charge a fee for this Package itself. However, you may distribute this Package in aggregate with other (possibly commercial) programs as part of a larger (possibly commercial) software distribution provided that you do not advertise this Package as a product of your own. You may embed this Package's interpreter within an executable of yours (by linking); this shall be construed as a mere form of aggregation, provided that the complete Standard Version of the interpreter is so embedded.
6. The scripts and library files supplied as input to or produced as output from the programs of this Package do not automatically fall under the copyright of this Package, but belong to whoever generated them, and may be sold commercially, and may be aggregated with this Package. If such scripts or library files are aggregated with this Package through the so-called "undump" or "unexec" methods of producing a binary executable image, then distribution of such an image shall neither be construed as a distribution of this Package nor shall it fall under the restrictions of Paragraphs 3 and 4, provided that you do not represent such an executable image as a Standard Version of this Package.
7. C subroutines (or comparably compiled subroutines in other languages) supplied by you and linked into this Package in order to emulate subroutines and variables of the language defined by this Package shall not be considered part of this Package, but are the equivalent of input as in Paragraph 6, provided these subroutines do not change the language in any way that would cause it to fail the regression tests for the language.
8. Aggregation of this Package with a commercial distribution is always permitted provided that the use of this Package is embedded; that is, when no overt attempt is made to make this Package's interfaces visible to the end user of the commercial distribution. Such use shall not be construed as a distribution of this Package.
9. The name of the Copyright Holder may not be used to endorse or promote products derived from this software without specific prior written permission.
10. THIS PACKAGE IS PROVIDED "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTIBILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The End

PHP

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from PHP. Under the terms of the PHP license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the

Oracle program, including the PHP software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the PHP software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or PHP.

The PHP License

```
The PHP License, version 3.0
Copyright(c) 1999-2004 The PHP Group. All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, is permitted provided that the following conditions
are met:

  1. Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

  2. Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

  3. The name "PHP" must not be used to endorse or promote products
     derived from this software without prior written permission. For
     written permission, please contact group@php.net.

  4. Products derived from this software may not be called "PHP", nor
     may "PHP" appear in their name, without prior written permission
     from group@php.net.  You may indicate that your software works in
     conjunction with PHP by saying "Foo for PHP" instead of calling
     it "PHP Foo" or "phpfoo"

  5. The PHP Group may publish revised and/or new versions of the
     license from time to time. Each version will be given a
     distinguishing version number.
     Once covered code has been published under a particular version
     of the license, you may always continue to use it under the terms
     of that version. You may also choose to use such covered code
     under the terms of any subsequent version of the license
     published by the PHP Group. No one other than the PHP Group has
     the right to modify the terms applicable to covered code created
     under this License.

  6. Redistributions of any form whatsoever must retain the following
     acknowledgment:
     "This product includes PHP, freely available from
     <http://www.php.net/>".

THIS SOFTWARE IS PROVIDED BY THE PHP DEVELOPMENT TEAM ``AS IS'' AND
ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE PHP
DEVELOPMENT TEAM OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT,
INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
OF THE POSSIBILITY OF SUCH DAMAGE.
```

FastCGI

Oracle is required to provide the text of the third-party license, but the third-party program will be subject to the Oracle license, and Oracle will NOT provide warranties and technical support for the third-party technology.

This program contains third-party code from FastCGI. Under the terms of the FastCGI license, Oracle is required to provide the following notices. Note, however, that the Oracle program license that accompanied this product determines your right to use the Oracle program, including the FastCGI software, and the terms contained in the following notices do not change those rights. Notwithstanding anything to the contrary in the Oracle program license, the FastCGI software is provided by Oracle "AS IS" and without warranty or support of any kind from Oracle or FastCGI.

FastCGI Developer's Kit License

This FastCGI application library source and object code (the "Software") and its documentation (the "Documentation") are copyrighted by Open Market, Inc ("Open Market"). The following terms apply to all files associated with the Software and Documentation unless explicitly disclaimed in individual files.

Open Market permits you to use, copy, modify, distribute, and license this Software and the Documentation solely for the purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions.

No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this Software and Documentation may be copyrighted by their authors and need not follow the licensing terms described here, but the modified Software and Documentation must be used for the sole purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose. If modifications to this Software and Documentation have new licensing terms, the new terms must protect Open Market's proprietary rights in the Software and Documentation to the same extent as these licensing terms and must be clearly indicated on the first page of each file where they apply.

Open Market shall retain all right, title and interest in and to the Software and Documentation, including without limitation all patent, copyright, trade secret and other proprietary rights.

OPEN MARKET MAKES NO EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE SOFTWARE OR THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL OPEN MARKET BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY DAMAGES ARISING FROM OR RELATING TO THIS SOFTWARE OR THE DOCUMENTATION, INCLUDING, WITHOUT LIMITATION, ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR SIMILAR DAMAGES, INCLUDING LOST PROFITS OR LOST DATA, EVEN IF OPEN MARKET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS". OPEN MARKET HAS NO LIABILITY IN CONTRACT, TORT, NEGLIGENCE OR OTHERWISE ARISING OUT OF THIS SOFTWARE OR THE DOCUMENTATION.

Module mod_fastcgi License

This FastCGI application library source and object code (the "Software") and its documentation (the "Documentation") are copyrighted by Open Market, Inc ("Open Market"). The following terms apply to all files associated with the Software and Documentation unless explicitly disclaimed in individual files.

Open Market permits you to use, copy, modify, distribute, and license this Software and the Documentation solely for the purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose, provided that existing copyright notices are retained in all copies and that this notice is included verbatim in any distributions.

No written agreement, license, or royalty fee is required for any of the authorized uses. Modifications to this Software and Documentation may be copyrighted by their authors and need not follow the licensing terms described here, but the modified Software and Documentation must be used for the sole purpose of implementing the FastCGI specification defined by Open Market or derivative specifications publicly endorsed by Open Market and promulgated by an open standards organization and for no other purpose. If modifications to this Software and Documentation have new licensing terms, the new terms must protect Open Market's proprietary rights in the Software and Documentation to the same extent as these licensing terms and must be clearly indicated on the first page of each file where they apply.

Open Market shall retain all right, title and interest in and to the Software and Documentation, including without limitation all patent, copyright, trade secret and other proprietary rights.

OPEN MARKET MAKES NO EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE SOFTWARE OR THE DOCUMENTATION, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL OPEN MARKET BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY DAMAGES ARISING FROM OR RELATING TO THIS SOFTWARE OR THE DOCUMENTATION, INCLUDING, WITHOUT LIMITATION, ANY INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES OR SIMILAR DAMAGES, INCLUDING LOST PROFITS OR LOST DATA, EVEN IF OPEN MARKET HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE SOFTWARE AND DOCUMENTATION ARE PROVIDED "AS IS". OPEN MARKET HAS NO LIABILITY IN CONTRACT, TORT, NEGLIGENCE OR OTHERWISE ARISING OUT OF THIS SOFTWARE OR THE DOCUMENTATION.

Glossary

Apache

Apache is a public domain HTTP server derived from the National Center for Supercomputing Applications (NCSA).

authentication

The process of verifying the identity of a user, device, or other entity in a host system, often as a prerequisite to granting access to resources in a system. A recipient of an authenticated message can be certain of the message's origin (its sender). Authentication is presumed to preclude the possibility that another party has impersonated the sender.

availability

The percentage or amount of scheduled time that a computing system provides application service.

CA

See **certificate authority**.

certificate

Also called a **digital certificate**. An ITU x.509 v3 standard data structure that securely binds an identity to a public key.

A certificate is created when an entity's public **key** is signed by a trusted identity, a **certificate authority** The certificate ensures that the entity's information is correct and that the public key actually belongs to that entity.

A certificate contains the entity's name, identifying information, and public key. It is also likely to contain a serial number, expiration date, and information about the rights, uses, and privileges associated with the certificate. Finally, it contains information about the certificate authority that issued it.

certificate authority

A trusted third party that certifies that other entities—users, databases, administrators, clients, servers—are who they say they are. When it certifies a user, the certificate authority first seeks verification that the user is not on the certificate revocation list (CRL), then verifies the user's identity and grants a certificate, signing it with the certificate authority's private **key**. The certificate authority has its own certificate and public key which it publishes. Servers and clients use these to verify signatures the certificate authority has made. A certificate authority might be an external company that offers certificate services, or an internal organization such as a corporate MIS department.

CGI

Common Gateway Interface (CGI) is the industry-standard technique for transferring information between a Web server and any program designed to accept and return data that conforms to the CGI specifications.

ciphertext

Data that has been encrypted. Cipher text is unreadable until it has been converted to plain text (decrypted) with a key. See **decryption**.

cipher suite

A set of authentication, encryption, and data integrity algorithms used for exchanging messages between network nodes. During an SSL handshake, for example, the two nodes negotiate to see which cipher suite they will use when transmitting messages back and forth.

cleartext

See **plaintext**.

cryptography

The art of protecting information by transforming it (encrypting) into an unreadable format. See **encryption**.

DAD

See **database access descriptor**.

database access descriptor

A database access descriptor (DAD) is a set of values that specify how an application connects to an Oracle database to fulfill an HTTP request. The information in the DAD includes the username (which also specifies the schema and the privileges), password, connect-string, error log file, standard error message, and national language support (NLS) parameters such as NLS language, NLS date format, NLS date language, and NLS currency.

decryption

The process of converting the contents of an encrypted message (**ciphertext**) back into its original readable format (**plaintext**).

DES

Data Encryption Standard. A commonly used symmetric **key encryption** method that uses a 56-bit key.

de-militarized zone

A de-militarized zone (DMZ) is a set of machines that are isolated from the internet by a firewall on one side, and from a company's intranet by a firewall on the other side. This set of machines are viewed as semi-secure. They are protected from the open internet, but are not completely trusted like machines that are inside the second firewall and part of the company's intranet. In a typical application server setup with a DMZ, only the Web listener and the static content for the Web site are placed in the DMZ. All business logic, databases, and other critical data and systems in the intranet are protected.

Diffie-Hellman key negotiation algorithm

Diffie-Hellman key negotiation algorithm is a method that lets two parties communicating over an insecure channel to agree upon a random number known only to them. Though the parties exchange information over the insecure channel during execution of the Diffie-Hellman key negotiation algorithm, it is computationally infeasible for an attacker to deduce the random number they agree upon by analyzing their network communications. Oracle Advanced Security uses the Diffie-Hellman key negotiation algorithm to generate session keys.

digital certificate

See **certificate**.

digital wallet

See **wallet**.

<Directory>

It is used to enclose a group of directives that apply only to the named directory and subdirectories of that directory. Any directory that is allowed in a directory context may be used. The directory is either the full path to a directory, or a wildcard string. In a wildcard string, ? matches any single character and * matches any sequences of characters. It is important to note that `<Directory />` operated on the whole file system, where as `<Directory` *`dir`*`>` refers to absolute directories. `<Directory>` containers cannot be nested inside each other, but can refer to directories in the document root that are nested.

<DirectoryMatch>

It should be used when specifying regular expressions, instead of using the tilde form of `<Directory>` with wildcards in the directory specification. The following two examples have the same result, matching directories starting with `web` and ending with a number from 1 to 9:

```
<Directory ~/web[1-9]/>
<DirectoryMatch "/web[1-9]/">
```

directory information tree

A hierarchical tree-like structure consisting of the DNs of the directory entries. See **distinguished name**.

distinguished name

The unique name of a directory entry. It comprises all of the individual names of the parent entries back to the root in the **directory information tree**.

DIT

See **directory information tree**.

DMZ

See **de-militarized zone**.

DN

See **distinguished name**.

encryption

The process of disguising a message thereby rendering it unreadable to any but the intended recipient. Encryption is performed by translating data into secret code. There are two main types of encryption: **public-key encryption** (or asymmetric-key encryption) and symmetric-key encryption.

entry

In the context of a directory service, entries are the building blocks of a directory. An entry is a collection of information about an object in the directory. Each entry is composed of a set of attributes that describe one particular trait of the object. For example, if a directory entry describes a person, that entry can have attributes such as first name, last name, telephone number, or e-mail address.

failover

The ability to reconfigure a computing system to utilize an alternate active component when a similar component fails.

<Files>

The `<Files` *`file`*`>` and `</Files>` directives support access control by filename. It is comparable to the **<Directory>** and **<Location>** directives. The directives given within this section can be applied to any object within a base name (the last component of the filename) matching the specified file name. `<Files>` sections are processed in the order that they appear in the configuration file, after the `<Directory>` sections, and `.htaccess` files are read, but before `<Location>` sections. Note that the `<Files>` directives can be nested inside `<Directory>` sections to restrict the portion of the file system to which they apply.

<FilesMatch>

Provides access control by filename, just as the **<Files>** directive does. However, it accepts regular expression.

HTTP

See **Hypertext Transfer Protocol**.

Hypertext Transfer Protocol

Hypertext Transfer Protocol (HTTP) is the underlying format used by the Web to format and transmit messages and determine what actions Web servers and browsers should take in response to various commands. HTTP is the protocol used between Oracle Application Server and clients.

key

A password or a table needed to decipher encoded data.

Keystore

Keystore is a protected database that holds **key**s and **certificate**s for an enterprise. Access to a keystore is guarded by a password (defined at the time the keystore is created, by the person who creates the keystore, and changeable only when providing the current password).In addition, each **private key** in a keystore can be guarded by its own password.

Keytool

Keytool is a **key** and **certificate** management utility.

LDAP

See **Lightweight Directory Access Protocol**.

Lightweight Directory Access Protocol

A standard, extensible directory access protocol. It is a common language that **LDAP** clients and servers use to communicate. The framework of design conventions supporting industry-standard directory products, such as the Oracle Internet Directory.

<Limit>

`<Limit` *`method`*`>` defines a block according to the HTTP method of the incoming request.

<LimitExcept>

Restrict access controls to all HTTP methods except the named ones.

<Location>

Limits the application of the directives within a block to those URLs specified, rather than to the physical file location like the **<Directory>** directive. `<Location>` sections are processed in the order that they appear in the configuration file, after the `<Directory>` sections and `.htaccess` files are read, and after the **<Files>** sections. `<Location>` accepts wildcard directories and regular expressions with the tilde character.

<LocationMatch>

Functions in an identical manner to **<Location>** and you should use it for specifying regular expressions instead of the tilde form of `<Location>` with wildcards in the location specification.

MD5

A hashing algorithm intended for use on 32-bit machines to create digital signatures. MD5 is a **one-way hash function**, meaning that it converts a message into a fixed string of digits that form a **message digest**.

message digest

Representation of text as a string of single digits. It is created using a formula called a **one-way hash function**.

modules

Modules extend the basic functionality of the Web server and support integration between Oracle HTTP Server and other Oracle Application Server components.

one-way hash function

An algorithm that turns a message into a single string of digits. "One way" means that it is almost impossible to derive the original message from the string of digits. The calculated **message digest** can be compared with the message digest that is decrypted with a **public key** to verify that the message has not been tampered with.

OPMN

See **Oracle Process Manager and Notification Server**.

Oracle Process Manager and Notification Server

Oracle Process Manager and Notification Server (OPMN) manages Oracle HTTP Server and OC4J processes within an application server instance. It channels all events from different components to all components interested in receiving them.

PEM

Privacy-Enhanced Electronic Mail. An **encryption** technique that provides encryption, authentication, message integrity, and **key** management.

plaintext

Also called cleartext. Unencrypted data in ASCII format.

port

A port is a number that TCP uses to route transmitted data to and from a particular program.

private key

In **public-key cryptography**, this **key** is the secret key. It is primarily used for decryption but is also used for encryption with digital signatures. See **public/private key pair**.

public key

In **public-key cryptography**, this key is made public to all. It is primarily used for encryption but can be used for verifying signatures. See **public/private key pair**.

public-key cryptography

Encryption method that uses two different random numbers (**key**s). See **public key** and **public-key encryption**.

public-key encryption

The process where the sender of a message encrypts the message with the public **key** of the recipient. Upon delivery, the message is decrypted by the recipient using its private key.

public/private key pair

A set of two numbers used for **encryption** and **decryption**, where one is called the **private key** and the other is called the **public key**. Public **key**s are typically made widely available, while private keys are held by their respective owners. Though mathematically related, it is generally viewed as computationally infeasible to derive the private key from the public key. Public and private keys are used only with asymmetric encryption algorithms, also called **public-key encryption** algorithms, or public-key cryptosystems. Data encrypted with either a public key or a private key from a key pair can be decrypted with its associated key from the key-pair. However, data encrypted with a public key cannot be decrypted with the same public key, and data encrypted with a private key cannot be decrypted with the same private key.

RSA

A **public-key encryption** technology developed by RSA Data Security. The RSA algorithm is based on the fact that it is laborious to factor very large numbers. This makes it mathematically unfeasible, because of the computing power and time required to decode an RSA **key**.

scalability

A measure of how well the software or hardware product is able to adapt to future business needs.

SHA

See **Secure Hash Algorithm**.

Secure Hash Algorithm

Secure Hash Algorithm assures data integrity by generating a 160-bit cryptographic message digest value from given data. If as little as a single bit in the data is modified, the Secure Hash Algorithm checksum for the data changes. Forgery of a given data set in a way that will cause the Secure Hash Algorithm to generate the same result as that for the original data is considered computationally infeasible.

An algorithm that takes a message of less than 264 bits in length and produces a 160-bit message digest. The algorithm is slightly slower than MD5, but the larger message digest makes it more secure against brute-force collision and inversion attacks.

Secure Shell

Secure Shell (SSH) is a well known protocol and has widely available implementation that provide a secure connection tunneling solution, very similar to what port tunneling offers. SSH provides a daemon on both the client and server sides of a connection. Clients connect to the local daemon rather than connecting directly to the server. The local SSH daemon then establishes a secure connection to the daemon on the server side. Communication is then routed from the client, through the client side daemon to the server side daemon and then on to the actual server. This allows a client/server program that uses an insecure protocol to be tunneled through a secure channel. For our purposes, the disadvantage of SSH is that it requires two hops to occur and that the implementations available do not perform and scale well enough. More information on SSH can be obtained from

```
http:www.ssh.org
```

Secure Sockets Layer

Secure Sockets Layer (SSL) is a standard for the secure transmission of documents over the Internet using HTTPS (secure HTTP). SSL uses digital signatures to ensure that transmitted data is not tampered with.

single sign-on

Single sign-on enables a you to authenticate once, combined with strong authentication occurring transparently in subsequent connections to other databases or applications. It lets you access multiple accounts and applications with a single password, entered during a single connection.

SSL

See **Secure Sockets Layer**.

SSH

See **Secure Shell**.

<VirtualHost>

Oracle HTTP Server has the capabilities to serve many different Web sites simultaneously. Directives can also be scoped by placing them inside

`<VirtualHost>` sections, so that they will only apply to requests for a particular Web site.

Virtual host refers to the practice of maintaining more than one server on one machine, as differentiated by their apparent hostname. For example, it is often desirable for companies sharing a Web server to have their own domain, and Web servers accessible as, for example, `www.oracle1.com` and `www.oracle2.com`, without requiring you to know any extra path information.

wallet

Also called a **digital wallet**. A wallet is a data structure used to store and manage security credentials for an individual entity. It implements the storage and retrieval of credentials for use with various cryptographic services. A **Wallet Resource Locator** (WRL) provides all the necessary information to locate the wallet.

Wallet Resource Locator

A wallet resource locator (WRL) provides all necessary information to locate a wallet. It is a path to an operating system directory that contains a wallet.

WRL

See **Wallet Resource Locator**.

X.509

Public **key**s can be formed in various data formats. The X.509 v3 format is one such popular format.

Index